Broadcast Searchable Keyword Encryption

Mohamed Ali Hamza Ali  
School of Computer Science and Engineering,  
University of Electronic Science and Technology of China  
mody_231279@yahoo.com

Fagen Li  
School of Computer Science and Engineering,  
University of Electronic Science and Technology of China  
fagenli@uestc.edu.cn

Ting Zhong  
Information Security Dept, School of Computer Science,  
University of Electronic Science and Technology of China  
zhongting@uestc.edu.cn

Zhiguan Qin  
School of Computer Science and Engineering,  
University of Electronic Science and Technology of China  
qinzg@uestc.edu.cn

Ahmed Abdelrahaman A. A  
Department of Electrical and Computer Engineering, Karary University, Omdurman, Sudan  
mogwari2000@yahoo.com

Abstract—Broadcast Searchable Keywords Encryption (BSKE) is a novel scheme that allows searching in encrypted data without knowing a secret key. Consider Bob wants to encrypt the same data under master public key for a group of users and stores this encrypted data with Alice, Malice is one of those recipients he asks Alice using his private key whether or not she has stored encrypted data, then Alice will search in all encrypted data using the master public key either she finds a matching then returns encrypted data to Malice or does nothing. We hopefully wish Alice will not learn anything from both encrypted data and the query. Our scheme contributes by providing fixed ciphertext $O(3)$ which does not grow with the number of recipients, efficient revocable for users without resets security parameters and the computational cost in the cloud side only. BSKE is designed based on random oracle model and the scheme is secured against adaptive chosen keyword attack.

Keywords— Broadcast encryption, searchable keywords encryption, data cloud privacy, outsourcing application

I. INTRODUCTION

Security of cloud computing application necessitates concealing the information from the view of a cloud provider that is because a data owner aims to hide his critical information from the cloud providers, who are considered as untrusted party.

Data owner intends to secure his information by applying any encryption algorithms (confidentiality) and access control list (authorization) before outsourced in cloud, and so broadcast encryption algorithms can achieve both two tasks (confidentiality and authorization).

The challenge is when an authorized user (questioner) wishes to explore those encrypted data on the cloud without revealing his question, then the cloud searches among all encrypted data, thus searchable keyword encryption algorithm can accomplish this challenge.

Public key searchable encryption (PKSE) can provide privacy for all parties without reveal any information to the cloud [1].

Whilst public-key broadcast encryption (BE) is the way to broadcast encrypted data to all authorized users which belongs to the set $S$ using a unique public key. Any user in $S$ can decrypt the message by using his own private key while users outside $S$ should not be able to do so even if all of them are colluded. Such a scheme has been motivated by many applications like pay-TV systems and the distribution of copyrighted materials as well as cloud computing [2].

Some application requires to send the same encrypted data for a number of users, by applying public key Encryption with keyword encryption (PEKS) [1] it can generate different ciphertexts for different users which leads to increases in the size of database proportionally with the number of recipients, induces Inflating of data issues [3], this increases the traffic of packets over network, more computational cost in the client sides (bilinear algorithm) and replicates encrypted data infecting security.

Our proposed scheme Broadcast Searchable Keyword Encryption (BSKE) can be applied to encrypt files, which enables sharing a file among privileged users over a public server or cloud provider. An encrypted data can be created by anyone using the public key. The authorized set can be determined by the creator of the data (broadcaster).

Public-key (BKSE) can be used in case a user wants to retrieve only those information that contain a particular interested keyword, among the huge number of information in which the user is specified as a privileged user, but without giving the server ability to know anything of encrypted data. BKSE enables authorized user to ask the server (by a
trapdoor) to check whether it has a specific keyword W or not. In this way the server should learn nothing about the information. All other users outside the privileged set S cannot learn anything, in particular, and also cannot generate such a valid trapdoor, even if they are colluded [4].

Another application where we can apply using our scheme is secured gateway application such as sending email for the group, in which any user can encrypt some documents with master public key of the scheme and also create BKSE of keyword e.g. (subject,” top secret”, “urgent”...etc.) with master public key, then receiver can preconfigure the router to route the encrypted data to certain path (firewall, IPSec, antivirus, send SMS in mobile,...etc) [5] according to some specifications from the users by using their own private keys (trapdoors), when the encrypted packet (with BKSE) arrives at the gateway it is matched with all stored trapdoors if it finds a match then it will route according of pre-setting, we wish that the gateway does not learn anything [6].

The cloud will store the encrypted keyword from data owner that send for many users in storage with only one BKSE. This technique can reduce the size of the server storage from O(2n) (if we use PKSE) to O(3) where n is number of recipients, because in the other schemes we need to create for each user different PEKS for same keyword and the total is 2n of PEKS and it consumes the storage of cloud and also add loading of traffic of communication between the data owner and the cloud sever, that causes more expenses (The average storage costs per month $0.25/GB/Month). [7].

However, there was still an open problem that is, how to encrypt the data and broadcasts it for all members in the group set S, which allows any user to ask query about encrypted keyword using his/her private key (trapdoors). The server cloud responds by matching the trapdoor request with all BKSE that are stored in a specific storage. If the server finds a matching it will return the encrypted data and send it back to the intended client without learning anything about the keyword from both BEKS and trapdoor.

II. APPLICATION

The proposed scheme can be applied in many aspects of information security technology especially in outsourcing applications when storing data with an untrusted third-party, however there are no obstacles if the data did not need confidentiality so it’s easy to query database server whether or not it had the information, but when the data are encrypted and without knowing the secret key it is impossible to fetch this data, so many applications can use this scheme in secured cloud storage application the data owner can encrypt and store it in cloud storage and enables a number of users to ask and query for encrypted data, obviously he also can ask the cloud in future to retrieve the encrypted data, there are many aspects of the application as following areas:

- Geographical information system (GIS) the map creator is intended to encrypt the maps (sensitive data) and stores all encrypted maps in untrusted server after enabling some authorized users to browse the maps and asking the server by coordinates of the target[9].
- Domain Name System Security Extensions (DNSSEC) enables security for Domain name system which provides authentication and data integrity but cannot provide confidentiality [10]. This is because when we try to encrypt the domain names or IP address is hard for DNS server to lookup in the database and get IP address so we can resolve this by applying our scheme and encrypt the domain names database with broadcast searchable encryption and allows DNS server search anonymously for IP addresses or Domain names without revealing which domain the client wants to browse or which IP address in this way we can protect the website from being attacked.
- Access control and file sharing systems.
- Group emailing system and social network applications.
- Secure indexing application can map searchable encrypted keyword as an index of the block ,and encrypted that contains this block with any general broadcast encryption, so outsourcing server will not know the index number and data block information.
- In general for all outsourcing database in an untrusted server like(searchable encrypted Image, Secure Archive system et al).

III. RELATED WORK

Proposed scheme focuses on privacy in database generally and specially cloud computing, the aim is to reduce the size of encrypted keyword in the database and reduce computational cost from the side of client.

Using broadcast public searchable encryption is to encrypt one keyword for many users in a group to avoid duplication of data as in previous schemes.

Nuttapong et al proposed the Forward Secure Searchable Broadcast Encryption their scheme is based on Hierarchical Identity, they focused on forwarding security by allowing the users to change their private keys at certain intervals while the public key is fixed, the size of ciphertext and private key are $O(\log^2 T)$ where T is the number of private keys cycles [11].

Boneh and Giovanniel constructed Public Key Encryption with keyword Search in this scheme sender first encrypts the file or the message with public key of receiver or the questioner and sends the cipher text to server or to the gateway.

The recipient asks for particular keyword by using his private key by computing the trapdoor and sent it to the database server or gateway, and then database server will search for all encrypted keywords and check if it exists or not. The size of ciphertext is O(2) [1].

Boneh et al proposed Collusion Resistant Broadcast Encryption with Short Ciphertexts O (1) and Private Keys O (1) [12]. Our scheme is based on it because of short size of...
ciphertext and private key but still the size of public key is large O(n).

A. Fiat et al. Built t-Collusion resistant schemes broadcast encryption the authors designed scheme that can resist t users colluded together, the size of ciphertext is O(t2-log n) and the size private key is O(t-log n)[13]. Our proposed scheme didn’t choose this algorithm to construct scheme.

Broadcast to large sets D, M. Naor, and J. Lotspiech el designed Revocation and tracing schemes for stateless receivers scheme with ciphertext size = O(r) where r is number of revoked users and private key size=O (log n) its Useful if small number of revoked players. Still the size of ciphertext and private key is large and not stable [14].

Eceli and Delerabl build Identity-Based Broadcast Encryption with Constant Size Ciphertexts O (2) and Private Keys O (1) [15] this first Identity Based broadcast Encryption with short size but has no method of revocation techniques.

IV. PRELIMINARIES

The new proposed scheme, Broadcast Searchable Keywords Encryption (BSKE), consists of four algorithms which are declared in the following.

Keygen (κ, n) is the key generation algorithm in which the input κ security parameter and the maximum number of users (n). It generates the master public key PK and also computes private keys for all users \{d_1, ..., d_n\}

BSKE (W, PK, S): This algorithm computes the encrypted keyword. First the data owner inputs the group set S ⊆ {1, ..., n}, master public key PK and the keyword W. algorithm outputs BSKE which consists of three parts (C_0, C_1, C_2).

Trapdoor (d_k, s, W): This function is executed by the legitimate client who inputs the keyword W, the private key (d_k), the k index of the user and the group S then it computes the trapdoors \langle T_0, T_1, T_2 \rangle . Later the output of this function will be sent to the server as query request.

Test (PK, S, BEKS, T_0): This function is used by the cloud server to examine the validity of the BSKE and trapdoor query. The cloud server will search and match all BSKE in the encrypted database on trapdoors. Then it responds by yes or no accordingly.

A. Bilinear mapping

This scheme is based on the concept of bilinear pairing maps [16], that is particularly described in Boneh and Franklin used elliptic curve [17] to build the bilinear mapping function with two groups by mapping to groups e: G×G → G_T. Both groups have the same prime order group; the map function must satisfy the following properties:

Computability: There is polynomial time algorithm by given g_1, g_2 ∈ G that can compute e(g_1, g_2) ∈ G_T.

Bilinearity: given any a, b ∈ Z_p the bilinearity function is such that e(g^a, g^b) = e(g, g)^{ab} ∈ G_T.

Non-Degenerate: g is generator of G then e(g, g) is generator of G_T leads e(g, g) ≠ 1.

B. Complexity Assumptions

Security of system is based on new a complexity assumption called the Bilinear Diffie-Hellman Summation Exponent Assumption (n-BDHSE). A symmetric pairing e: G×G → G_T where G is a bilinear multiplicative group of prime order p. GT is target group of prime order P. The problem n-BDHSE is given (S, h, g, g^a, g^a^2, ..., g^{a^{2n}}) ∈ G^{2n+3} as input of (2n+3)-tuple, output vector \vec{Y} ∈ G^{#S} such that

\[ e \left( \prod_{j \in S} g^{a^{j+1}}, h \right) = e \left( \prod_{j \in S} g^{a^{j+1-j+1}}, h \right) \]

Where #S is the number of elements in the set S with constraint \vec{Y} non-zeros vector, and k ≠ 0. An algorithm A has advantage A_{DHSE} = e to solve n-BDHSE problem in group e: G×G → G_T if

\[ pr(S, h, g, g^a, g^{a^2}, ..., g^{a^{2n}} = \vec{Y}) \]

Where probability is distributed over random selection generator g of G, random choice of a ∈ Z_p, random of choice T ∈ G_T and random bits selected by A, as shorthand g_i used to denote g_i = g^{a_i} ∈ G.

V. CONSTRUCTION

This section describes and builds the proposed scheme, which consists of four functions as follow:

A. Setup (κ, n):

The setup algorithm inputs security parameter κ and the number of users’ n and by using bilinear pairing function technique. Picks up a random generator g ∈ G and \alpha ∈ Z_p computes g_i = g^{\alpha_i} for i = 1, ..., 2n then selects a random \gamma ∈ Z_p and computes v = g^\gamma ∈ G the master public key for set S is PK = (v, g, g^a, g^{a^2}, ..., g^{a^{2n}}). And the private key for each user k ∈ S will be d_k = g^\gamma_v = v^{(\alpha_k)} ∈ G, choosing a cryptographic hash function H: [0, 1]^* → G.

Setup algorithm will send the private key for user in secure channel and publish the master public key note that the master public key for all user in the set S

B. BSKE (W, PK, S):

The sender (data owner) inputs the keyword W, master public key and authorizes set S then chooses a random t ∈ Z_p and computes \{g^t, (H(W))^t, (v \prod_{j \in S} g^{a^{j+1}})^t\} = BSKE and sends this data to the cloud server.
C. Trapdoor \((d_t, k, S, W)\):

The legitimate client asks the cloud server whether it stored the encrypted keyword by using his own private key and computes trapdoor of the keyword firstly the client chooses a random \(r \in \mathbb{Z}_p\) and computes

\[
g', \left( g^{(\alpha^r)} \right)^r, \left\{ d_k \left( H(W) \right), \prod_{j \in S} g^{x_{i+1-i-j}} \right\} = T_w
\]

This sends a tuple to the cloud for validation of the encrypted keyword \(W\).

D. Test \((PK, S, BSKE, T_W)\):

Server upon receiving of the query \(T_W = \{T_0, T_1, T_2\}\) it starts search for all \(BSKE = \{C_0, C_1, C_2\}\) and checks if

\[
e(T_2, C_0) = e(T_1, C_2)
\]

Then, cloud server returns the encrypted part to the client, otherwise \(\bot\), for correctness there is matching equation it performs as the following:

\[
e(T_2, C_0) = e(T_1, C_2) = e^\left( \left( d_k \left( H(W) \right) \prod_{j \in S} g^{x_{i+1-i-j}} \right)^r, g^r \right) = e^\left( \left( g^{(\alpha^r)} \right)^r, \prod_{j \in S} g^{x_{i+1-i-j}} \right) = e^\left( \prod_{j \in S} g^{x_{i+1-i-j}} \right) \cdot e^\left( H(W) \right)^r, g^r \]

VI. Revocability

It is able to add and revoke user \(x\) from the set of authorized group by updating \(C_2 = C_2 \cdot \left( g^{x_{i+1}} \right)^t\) for adding or for revoking set \(C_2 = C_2 / \left( g^{x_{i+1}} \right)^t\) notice updating take place only in the value of \(C_2\) which it stored in the cloud, also we can calculate \(r\) by using some PRF and store the IV in the cloud this is to avoid storing data in the client machines.

VII. Efficient

The users can cached \(v \prod_{j \in S} g^{x_{i+1-i-j}}\) for BSKE also cached \(d_k \prod_{j \in S} g^{x_{i+1-i-j}}\) in their memory and reuse it anytime; the most computational cost is in bilinear mapping function which is taken over by the cloud server. So from the view of clients all the processes are not heavy.

VIII. Security Model

A. Theorem I

The Broadcast Searchable Keyword Encryption (BSKE) is non-interactively semantic secure in random oracle model against a chosen keyword attack assuming that \(n\)-BDHSE holds \([18]\).

B. Proof :

Let an attacker \(A\) try to play with challenger \(C\) security game to break security of the scheme, first step \(C\) runs keyGen algorithm, selects number of users \(n\), creates authorized set \(S\), creates master public key \(PK\), generates private keys and sends the \(PK\) and \(S\) to \(A\).

Fig. 1. (security Model)

In phase 1 \(A\) selects any unauthorized user \(i \notin S\) which \(A\) aims to attack, chooses any keyword \(W\) adaptively and asks \(C\) to give him \(T_i(W)\) (same as decryption services) , then \(C\) responds by finding private key for unauthorized user \(i \notin S\) , computes \(T_i(W)\) and submits it to \(A\) and continues asking \(C\) for different keywords.

Challenge steps when \(A\) finished all his queries of trapdoors then intends to submits two equal length keywords \(W_0, W_1\) to \(C\), then tosses fair flop coin and chooses \(W_0\) and computes \(BSKE_2(W_0)\) by running \(BSKE\) algorithm for authorized set \(S\) and sends it to \(A\) as challenge.

Phase2 same as above with condition \(W\) not asked before. Then \(A\) output guessing of \(b\) as \(b\) and wins if \(b = b^*\).
C. Theorem 1

The Broadcast Searchable Keyword Encryption scheme BSKE is semantically secure against chosen keyword attack in random oracle model assuming n-BDHSE is hard problem.

Proof: let A is an active attacker has advantage $\epsilon$ to break n-BDHSE problem assuming A asks at most $q_T$ hash function queries for keyword $H(W)$ also A can ask at most $q_T$ for trapdoor queries with constraint $q_H \cdot q_T > 0$.

$C$ is a simulator and has a running time approximately as A’s running time, $C$ is constructed to break the scheme at least $\epsilon^2 = \epsilon I(e_{DHSE}, q_T)$ where $e$ is base of natural logarithm.

D. Setup (n)

Challenger C starts creating master public key $PK$ for n users, picks random $u \in Z_p$ and computes

$$v = g^u \prod_{j \in S} \left( g^{(a^{u+1-j})} \right)^{-1}, \ \ PK = (g, g^a, g^{(a^2)}, ..., g^{(a^{2n})}, v)$$

The private key for user $i \notin S$

$$d_i = g^a \prod_{j \in S} \left( g^{(a^{u+1-j})} \right)^{-1}$$

$$= \left( g^u \prod_{j \in S} \left( g^{(a^{u+1-j})} \right)^{-1} \right)^{a^i} = v^{a^i} = \nu_i$$

And sends $\nu_i$ to A

E. H-queries

Challenger C prepares H-list consists of $<W_i, h_j, X_i, y_j>$ to simulate hash function, when A asks hash of keyword $W$ of user $k$, Challenger responds as follow:

1. Search of all H-list if $W_i$ is found then, return $H(W) = h_k$ to A otherwise

2. Challenger picks random $(C_i, X_i)$ where $C_i \in \{0,1\}$, $X_i \in Z_p$
   
   If $C_i = 0$ computes $h_i = g^{X_i} \prod_{j \in S} \left( g^{(a^{u+1-j})} \right)$

   If $C_i = 1$ computes $h_i = \left( g^{X_i} \right)^{a^i} \prod_{j \in S} \left( g^{(a^{u+1-j})} \right)$

   Where $y_j \in Z_p$ is random.

3. Add $(W_i, h_i, X_i, y_j)$ to the list and send $H(W) = h_k$ to A.

F. Trapdoor queries phase1

A asks challenger to create trapdoor for the keyword $W$ and using private key $d_k$ of user $k \notin S$ and C responds as:

1. Challenger runs H-queries to get $H(W)$ from $<W_j, h_j, X_j, y_j>$ if $C_i = 1$ then aborts.

2. Otherwise challenger computes

   $C_i = 0, h_i = g^{X_j} \prod_{j \in S} \left( g^{(a^{u+1-j})} \right) = H(W)$.

3. C picks random $r \in Z_p$ and computes $T_w = (T_0, T_1, T_2)$ as:

   $T_0 = g^{r^{a^{(1-k)}}}$

   $T_1 = g^{r^{a^i}}$

   $T_2 = \left( g^{u} \cdot H(W)^{a^k} \right)^{r}$

   In real scheme the trapdoor of third part is

   $T_2 = \left( d_k \cdot H(W) \right)^{a^k} \cdot \prod_{j \in S} \left( g^{(a^{u+1-j})} \right)^{-1} \cdot H(W)^{a^k} \cdot \prod_{j \in S} \left( g^{(a^{u+1-j})} \right)^{-1}$

   $= \left( g^{u} \right)^{a^k} \cdot \left( H(W) \right)^{a^k} \cdot \prod_{j \in S} \left( g^{(a^{u+1-j})} \right)$

   $= \left( g^{u} \right)^{a^k} \cdot \left( H(W) \right)^{a^k} \cdot \prod_{j \in S} \left( g^{(a^{u+1-j})} \right)$

   $= \left( g^{u} \right)^{a^k} \cdot \left( H(W) \right)^{a^k} \cdot \prod_{j \in S} \left( g^{(a^{u+1-j})} \right)$

   Challenger sends $T_w$ to A and from A’s view it look same as original scheme.

G. Challenge

A submits two equal length keywords $W_0, W_1$ (not asked in trapdoor phase) to C for challenge phase and responds by:

1. runs H-list queries twice to obtain hash function of two keywords $H(W_0), H(W_1)$ of $W_0, W_1$ as $<W_i, h_i, X_i, y_j>$ and $<W_j, h_j, X_j, y_j>$ respectively

2. If $C_i = 0$ and $C_j = 0$ then aborts.

3. if $C_i = 1$ and $C_j = 1$ C tosses fair coin $b \in \{0,1\}$ and select $W_0$ or $W_1$ respectively

4. Otherwise select where $C_i$ or $C_j$ it equals to one.

5. From H-list

   $H(W_b) = h_{i,j} = \left( g^{X_i} \right)^{a^i} \prod_{j \in S} \left( g^{(a^{u+1-j})} \right)$

   C picks random $t \in Z_p$ and computes

   $BSKE(S, W_0, PK) = (C_0, C_1, C_2)$

   $= \left( g^{r}, \cdot H(W_0)^{r}, \left( g^{a_i} \right)^{r} \right)$

   Note that $v = g^u \prod_{j \in S} \left( g^{(a^{u+1-j})} \right)^{-1}$

   $v \prod_{j \in S} \left( g^{(a^{u+1-j})} \right) = g^u \prod_{j \in S} \left( g^{(a^{u+1-j})} \right)^{-1} \cdot \prod_{j \in S} \left( g^{(a^{u+1-j})} \right) = g^u$

   $C$ sends $\{C_0, C_1, C_2\}$ as challenge to A.

H. Phase2

A resumes asking challenger Trapdoor queries same like phase 1 with constraint that queried keyword not mentioned in challenge phase.
I. Guess

Eventually A outputs the guess $b'$ of $b$ and wins if $b = b'$. By applying simulation it founds the verifying results as:

$$\frac{e(T_2, C_0)}{e(T_1, C_2)} = e(T_0, C_1)$$

(1)

From the left part of the above equation

$$e(T_2, C_0) = e(T_1, C_2) = e((g^u)^{a^\gamma}, (g^{a^\gamma} y')^v)$$

$$e((g^u)^{a^\gamma}, (g^{a^\gamma} y')^v) = e((H(W))^\gamma, g')$$

$$= e\left(\prod_{j \in S} g^{x_j}, g'\right)$$

$$= e\left(\prod_{j \in S} g^{(a^\gamma x_j)}, g'\right)$$

$$= e\left(\prod_{j \in S} g^{a^\gamma x_j}, g'\right)$$

(2)

Hence

$$C_1 = (H(W_b))^\gamma = \left(\prod_{j \in S} g^{x_j}\right)^\gamma$$

$$T_0 = (g^\gamma)^{(a^\gamma + \gamma)}$$

Then right side part of equation test (1) is

$$e(T_0, C_1) = e\left(\prod_{j \in S} g^{x_j}, g'\right)$$

$$= e\left(\prod_{j \in S} g^{a^\gamma x_j}, g'\right)$$

(3)

And (3) is same as (4) and the summation looked as the BSKE scheme from view of $A$.

In case if the attacker $k \notin S$ he tries to impersonate some user identity to ask query from Eq (4) the attacker wins the game if it successfully computes the vector $Y$ such that

$$e\left(\prod_{j \in S} g^{a^\gamma x_j}, h_k\right) = e\left(\prod_{j \in S} g^{a^\gamma x_j}, h\right)$$

Where $h = g'$ and that holds iff $A$ can breaks n-BDHSE, the attacker can win the above game if the simulation did not abort and the adversary succeed to guess bit b, suppose there are three events as follow [1]:

E1: simulator will not abort in trapdoor queries phase.
E2: simulator will not abort in challenge phase.
E3: A did not ask for $(W_b, W_f)$ in trapdoor queries phase.
E4: guess $b = b'$ correctly.

The probability for the first event is $pr[E_1] \geq \frac{1}{e}$,

$$pr[E_2] \geq \frac{1}{q_f}, pr[E_3] \geq 2e / q_H$$

And

$$pr[E_4] \geq \frac{1}{2}$$

Hence

$$\epsilon' = pr[E_1] \cap pr[E_2] \cap pr[E_3] \cap pr[E_4]$$

$$\epsilon' = \epsilon / (e q_H q_f)$$

Since $C$ can’t break at least $\epsilon' = \epsilon / (e q_H q_f)$ and this required to proof above theorem.

IX. PERFORMANCE AND IMPLANTATION

The scheme is implemented by using MIRACL library which is C++ SDK and contains more than hundred routines cover all aspects of multi-precision arithmetic and Mathematical of Cryptography [19], bilinear pairing function is chosen Super-Singular cryptography [GF(2^M)] type-I[20], for hash function used AES 128 bits.

Table 1 shows the result of running four algorithms when it changes maximum number of users, the last two columns are the weight of BSKE algorithm = time(BSKE)/time(Test) and the weight of Trapdoor algorithm = time(Trapdoor)/time(Test) (respectively). the weight function is used to compare BSKE and Test or evaluate the complexity in the user machines (data owner) and Cloud poivider, which is 12.4% from cloud computation cost also the weight of Trapdoor is 12.9%, that means the complexity in the clients is very light comparing with cloud.

Figure 2 shows that the algorithms are not affected by maximum number of users except KeyGen which increases the complexity in linear and remind algorithms are not affected.

The three algorithms in figure 3 are plotted and excluded KeyGen, obviously it seems that the Test algorithm is most computational cost.

Fig. 2. (KeyGen is affected with NO of users)
Client users use BSKE and Trapdoors algorithms which are influenced by the number of authorized users in the set #S Proportionally figure 4, whereas test algorithm is not affected. To handle this problem we cache initial multiplications and we get fixed and small complexity (using cached bke and using cached trap), note that calculating of cached value is one time after revoke or add a user.

X. CONCLUSION

The proposed scheme has been providing searchable keyword encryption for multi-user with fixed size ciphertext O(3), the scheme has proved semantically secure against adaptive keyword adversary on random oracle model under n-BDHSE assumption, we provided less computational cost on clients side whereas little heavy in cloud side enhancing scheme by applied pre-computations by cached the multiplication, we allowed data owner to revoke expired user. we were opening some problem such as the size of master public key is large and grows linearly with number of total user and number of users are bounded.

Acknowledgment

We thank ms. Xiaofen Wang for her great assistance, also we appricate mr Abed Eldaieem for his guiding and comments.

References

[7] Cloud Storage Solution - Barracuda.com
[8] Security management in health using ISO27799:
[12] Boneh, Gentry, and Brent, “Collusion Resistant Broadcast Encryption With Short Ciphertexts and Private Keys
[14] Revocation and tracing schemes for stateless receivers D. Naor, M. Naor, and J. Lotspiech
[15] Ecile and Delerabl build Identity-Based Broadcast Encryption with Constant Size Ciphertexts