Volume 7, Issue 3, March 2017 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Aggregate Key Searchable Encryption: A Technique for Group Data Sharing on Cloud

Ajay Gawade*
P.G Student (Computer Engineering)
Terna Engineering College, Nerul,
Mumbai, Maharashtra, India

Rashmi Dhumal
Asst. Professor (Computer Engineering)
Terna Engineering College, Nerul,
Mumbai, Maharashtra, India

Abstract :- Nowadays, everyone uses cloud storage to store their data on the cloud, so that it can be access their data from anytime, anywhere via internet. Organizations also employ cloud storage to increase their revenue as well as reduced the maintenance cost of the system. However, the user doesn't have trust on the cloud service provider as he loses control over the data. To address a problem, user encrypts their data before uploading to public cloud storage. The efficiency of selectively sharing encrypted files with the many users' demands separates encryption keys to be used for search as well as encryption. However, the data owner requires distribute a large number of keys to users and those users need to submit a large number of keyword trapdoors to perform searches over the shared data. This requires an efficient management of encryption keys for shared data access. Hence, we propose an Aggregate Key Searchable Encryption technique to share many files with a single key. The data owner distributes a single key for sharing group of files to the user and user submits a single trapdoor to perform searches over the shared files.

Keywords - Cloud Storage, Searchable Encryption, Data Sharing, Trapdoor, Cloud security.

I. INTRODUCTION

Currently, usage of cloud storage has been increased day to day life. Many Organizations and users are using cloud storage due to various benefits over traditional storage. Cloud storage can provide big storage, data backup, better accessibility, rapid deployment and reduced overall storage cost. It has also reduced the cost of software's as well as maintenance of the expensive hardware. Most of the online services are based on the cloud storage and anyone can use these online services from anywhere and anytime. For example, Google offers an online service known as Google Drive that enables a user to upload and download files from Google Drive at anytime and anywhere.

Data sharing is a major functionality in cloud storage. For example, social networking sites can allow their friends to view a subset of their private photo, video, etc. An organization may allow his/her user access rights to particular data. But, the challenging problem is that how to efficiently share encrypted data via public cloud storage which greatly increase the security of cloud. First of all, data owner needs to be downloading the encrypted data from the cloud storage and decrypt them, and then send them to other people for sharing. But this approach inefficient, impractical as it loses the purpose of cloud storage. To overcome this problem, data owner should be able to give the access rights of the sharing data to other people so that they can access these data from the cloud storage directly. However, while enjoying the various services of cloud storage, users are also increasing security related problem. For Example, unauthorized user accesses your data or misbehaving by the cloud service provider can cause serious breaches of private data or business secrets. To address this problem, the data owner encrypts their data before uploading them to the cloud. But, it is difficult to search over encrypted data to the user and retrieve only selected data containing given keywords. To address this issue, searchable encryption technique is used to search over encrypted data. In this technique, a collection of files and an index of these files are encrypted with the potential keywords and upload them to the cloud. Then, the user will send the corresponding keyword trapdoor to the cloud for retrieving data with a matching keyword. But, implementing such technique for large scale application involving millions of users and billions of files may still practical issues for efficient management of encryption keys. Data sharing with many users usually demands separate encryption keys to be used for each file. For example, sharing selected files with particular department or sharing a photo with particular friends in a social networking application on a cloud drive. This implies that the large number of keys that need to securely distributed to user both for decryption as well as search of those files. In such situation, data owner not only distribute a large of keys to the user, but also securely stored as well as manage by the user in their device. In addition, User wants to perform a keyword search over shared files required to generate a large number of keyword trapdoors and submitted to the cloud. To overcome this problem, we require an efficient management of encryption keys for sharing data access. Hence, we propose an Aggregate Key Searchable Encryption Technique. In this technique, the data owner distributes a single key for sharing a number of files to the users and user submits a single trapdoor for performing search over those files.

This paper is organized as follows: Section II includes literature survey. It involves the work done by the various researchers in the field of searching over encrypted data. Section III includes problem definition. In Section IV, we

explained proposed methodology. Section V includes implementation and results of our proposed system. We explain the performance of our system in section VI. Finally, section VII is the conclusion and future scope of the work.

II. LITERATURE SURVEY

In the recent years, searching over encrypted data has become an important issue in cryptography and security. Client outsources their data to a third party server. But, the client doesn't have trust on it as he /she loses control over their data. Hence, the data confidentiality is an important issue in cloud security. To achieve the data confidentiality, data is stored in encrypted form on cloud storage. But, it is complicated to perform any operation on encrypted data. There are several techniques present in the literature for searching over encrypted data as follows:

2.1 Practical technique for search over encrypted data

D. Song et al. [1] were the first to propose a technique to perform search operations on encrypted data without any leak of information. They have used pseudo random function, pseudo random generator and sequential scan. This technique provides query isolation, controlled searching and hidden search query. Third party server cannot able to search for arbitrary keywords without the client permission and cannot learn about the plaintext than the search result. This technique is secure, practical and has no communication overhead. But, this scheme used sequential scan which is not sufficient if data size is large and also takes more time to search over a large number of files.

2.2 Searchable Symmetric Encryption

Although previous technique provides search over encrypted data but it is too slow for searching over large number of files. To overcome above problem, searchable symmetric encryption technique was proposed by R. Curtmola et al. [2]. Searchable symmetric encryption scheme use symmetric encryption which provides searching capabilities over encrypted data. This technique provides symmetric encryption with search ability is called as secured index. With the given a particular keywords, it return a particular files associated with it. In this technique, data owner encrypts its data and indexes and sends the secure index together with encrypted data to the server. To perform keyword search over shared data, user need to generates and submit a keyword trapdoor to the server. Then, server uses this keyword trapdoor to perform search over encrypted data and return matching keywords file. This technique is better and more secure than previous technique. But, this technique not suitable for large data size and support only single user.

2.3 Public Key Encryption with a keyword search

Previous technique used symmetric encryption to perform search over encrypted data. Consider a scenario where user wants to retrieve data for an existing file that he /she didn't store themselves. At such situation, it would be impossible unless he/she has a common secret key with the user who encrypted these data. Hence, to address this problem, Boneh et al. [3] proposed public key encryption with keyword search. In this technique, file is encrypted using public key for the user who wants to store it in the un-trusted server. But, the authorized user can generates keyword trapdoor using his private key to perform search over encrypted data. This technique provides better security than previous technique, but increase number of keys which makes a system difficult. However, this technique needs to support for keyword trapdoor to identify keyword alone and it supports only single user.

2.4 Multi-user searchable encryption

The previous techniques provide search over encrypted data but it supports only single user. That is, data owner can store encrypted data can only perform a search and retrieve data. In cloud environment, search over large data under multi-tenancy is a general scenario. In such situation, data owner would like to share number of files to authorized users and each user gives access right can perform keyword search over shared data. A lot of work has been done a multi-user searchable encryption. Such as, single key combined with access control [4], [5], [22]. Since, they all used single key combined with access control to accomplish the goal. In MUSE techniques are constructed based on broadcast encryption to achieve coarse-grained access control and share the file's using searchable encryption key to all users who can access it. The main issue in MUSE is, how to manage which users can access which files and how to decrease the number of shared keys and trapdoors is not considered.

2.5 Multi-key searchable encryption

In the multi user searchable encryption, the number of trapdoors is proportional to number of files. That is, user needs to submit a number of trapdoors to perform search on shared encrypted data. To address this problem, Popa et al. [6] proposed a multi-key searchable encryption technique allows a user to search over shared data using a single keyword trapdoor instead of a multiple keyword trapdoor. For example, Alice has n encrypted files with her key uk_A on the cloud server and each file encrypts under a key k_i for i= {1,....., n}. To allow the cloud server adjust the right trapdoor for each file with index i, Alice stored the some public information called delta (Δ_{uk_A,k_i}) on cloud server. This public information is relevant to both uk_A and k_i.When Alice wants to search for a keyword over all the files, she will use uk_A to compute a trapdoor for the keyword and send to the cloud server. Then cloud server can use Δ_{uk_A,k_i} to convert a keyword trapdoor under k_i and performs a traditional single key search with the new trapdoor. This scheme allows a user to provide a single keyword trapdoor to the server, but still allows the server to search for that trapdoor keyword in files with different keys.

2.6 Key aggregate encryption for data sharing

In the traditional approach, to share group of files with separate encryption keys with the same user, data owner requires to distribute all the keys to user. To address this problem, Chu et al. [7] proposed a key aggregate cryptosystem to reduce the number of distributed data encryption keys. This scheme allows the data owner to generate single aggregate key for the group of files to the user and user can decrypt all this files with the single aggregate key. To allow a group of files encrypted with separate keys to be decrypted with a single aggregate key, data owner encrypt a message not only public key but also identifier of each file. Then data owner use his/her master secret key to generate a single aggregate key for group of files. This aggregate key is used to decrypt all the shared files by the user. But, this technique doesn't provide search over encrypted data.

III. PROBLEM DEFINITION

Consider a situation, where the two employees of corporation would like to share some confidential business data using public cloud storage. For example, John wants to upload a huge collection of financial files to cloud storage for the manager of different department to review. Suppose, those files contain sensitive information that should only be accessed by valid users. Consider Bob is one of the managers to view files related to his department. Due to the potential data leakage in the cloud such as data access by unauthorized users or misbehaving by the cloud operator can cause serious breaches of business secrets. So, John encrypts these files with different keys, and generates keyword cipher-texts based on the department names, before uploading to cloud storage. Then, John uploads and shared those files with managers using sharing functionality of the cloud storage. Now, Bob wants to view the files related to his department, John must delegate rights to Bob both for keyword search over those files and decryption rights for files related to Bob's department. With a conventional method, John must securely send all the searchable encryption keys to Bob. After receiving these keys, Bob must store them securely, and then he must generate all the keyword trapdoors using those keys in order to perform a keyword search. However, such technique required secure communication, storage as well as computational complexity which are clearly inefficient and impractical. That is, there is a need to manage number of encryption keys for shared data access. Hence, we propose an Aggregate Key Searchable Encryption technique to share multiple files with single key. The data owner distributes a single key for sharing large number of files to the user and user submits a single trapdoor for querying the shared files.

IV. PROPOSED METHODOLOGY

Existing techniques are used to single key to search the file either for single user or many users. But, there is a need of an efficient searching technique in which data owner distribute a single key for sharing a group of files and valid users submit a single trapdoor in order to perform keyword search over the shared files.

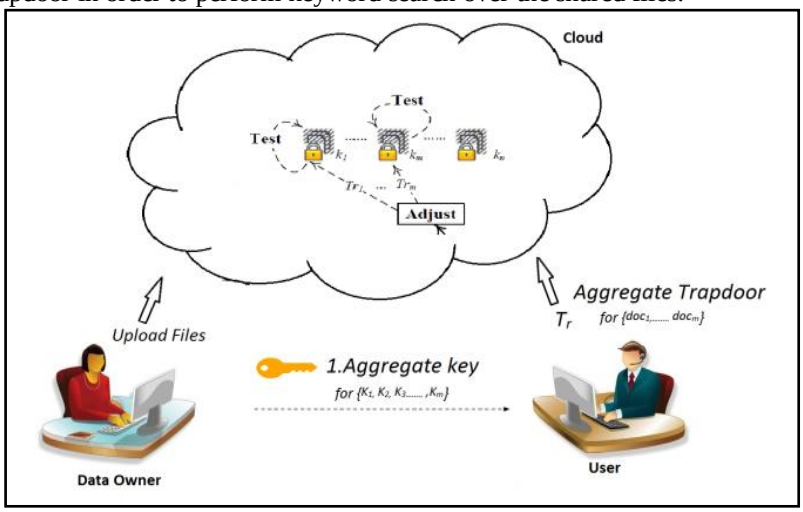

Figure 4.1 Architecture of proposed system

Hence, we propose an Aggregate Key Searchable Encryption technique to search multiple files using single aggregate key. As shown in Fig.4.1, data owner distribute a single aggregate key for sharing group of files with user and user submits a single aggregate trapdoor to the cloud server. The cloud server can use this aggregate trapdoor and some public information to perform keyword search and return the result to user. Since, the delegation of keyword search right can be achieved by sharing the single aggregate key in our system.

4.1 Bilinear pairing

Let G and G_1 be two cyclic groups of prime order p. Let g be a generator of G. A bilinear map or bilinear pairing e is then an efficiently computable function e: G x $G \rightarrow G_1$ satisfying the following properties:

1. Bi-linearity : For all u, v ∈ G and a, b ∈ Z_p , then
$$e(u^a, v^b) = e(u, v)^{ab}$$
2. Non-degeneracy: If g is a generator of G, then $e(g, g)$ is a generator of G_1.
3. Computability :There is an efficient algorithm to compute $e(u, v)$ for any u,v ∈ G

4.2 Aggregate Key Searchable Encryption Scheme

1) Setup *(1^λ, n):* this algorithm used by the cloud server to initialize system parameters as follows.

- Generate a bilinear map group system $B= (p, G, G_1, e(\cdot, \cdot))$, where p is the order of G and $2^\lambda \le p \le 2^{\lambda+1}$.
- Set n as the maximum possible number of files which belongs to a data owner. Pick a random generator $g \in G$ and a random $\alpha \in Z_p$, and computes $g_i = g^{(\alpha^i)} \in G$ for $i = \{1, 2, \cdots, n, n+2, \cdots, 2n\}$.
- Select a one-way hash function H: $\{0, 1\}^* \rightarrow G$.

2) Keygen*:* data owner uses this algorithm to generate his/her key pair. It picks a random $\gamma \in Z_p$, and outputs:

$$pk = v = g^\gamma, msk = \gamma \quad \text{..... (1)}$$

3) Encrypt *(pk, i):* data owner uses this algorithm to encrypt data and generate its keyword cipher-texts. To generate the keyword cipher-texts, this algorithm takes as input the file index $i \in \{1,..., n\}$, and:

- Randomly picks a $t \in Z_p$, as the searchable encryption key k_i of this file. Generates a delta Δ_i for k_i by computing:

$$c_1 = g^t, c_2 = (v \cdot g_i)^t \quad \text{..... (2)}$$

- for a keyword w, outputs its cipher-text c_w as:

$$c_w = e(g, H(w))^t / e(g_1, g_n)^t \quad \text{..... (3)}$$

Where c_1, c_2 are public and can be stored in the cloud server.

4) Extract *(msk, S):* This algorithm use *by* data owner to generate an aggregate searchable encryption key. For any subset $S \subseteq \{1,...,n\}$ which contains the indices of files, this algorithm takes as input the owner's master-secret key msk and outputs the aggregate key k_{agg}.

$$k_{agg} = \prod_{j \in S} g^\gamma_{n+1-j} \quad \text{..... (4)}$$

To delegate the keyword search right to a user, data owner will send k_{agg} and the set *S* to the user.

5) Trapdoor *(k_{agg} , w):* the user uses this algorithm to generate the trapdoor to perform keyword search. For all files which are relevant to the aggregate key k_{agg} ,this algorithm generates the only one trapdoor Tr for the keyword w by computing: Then, the user sends *(Tr, S)* to the cloud server. By computing:

$$Tr = k_{agg} \cdot H(w) \quad \text{..... (5)}$$

6) Adjust *(params, i, S, Tr):* the cloud server uses this algorithm to produce the right trapdoor. For each file in the set S, this algorithm takes as input the system public parameters params, the file index $i \in S$ and the aggregate trapdoor *Tr*, outputs the right trapdoor Tr_i by computing:

$$Tr_i = Tr. \prod_{j \in S, j \neq i} g_{n+1-j+i} \quad \text{..... (6)}$$

Then, the cloud server will use Test algorithm to complete the keyword search.

7) Test *(Tr_i , i):* the cloud server uses this algorithm to perform keyword search over the i-th file. For the i-th file, this algorithm takes as input the adjusted trapdoor Tr_i , the Δ_i= *(c_1, c_2)* relevant to its searchable encryption k_i and the subset *S*, outputs true or false by judging:

$$c_w =^? = e(Tr_i , c_1) / e(pub, c_2) \quad \text{..... (7)}$$

Where $pub = \prod_{j \in S} g_{n+1-j}$.

4.3 Identity based encryption

Today, Email is the one of the very important methods of communication for any organization and individual. It's incredible how email has changed our professional and social life. However, current industry standards do not place an importance of email security; most of the emails are currently send the data in plain text over the Internet or other networks. Emails can be intercepted easily by others. It could be possible that all non-encrypted email sent over a network or stored at an email server can be read, copied or altered. Since, there is a strong need for secure email delivery over the internet.In the existing system, there is problem with sending an aggregate searchable encryption key as it is through mail. Because, it does not prevent email server administrators, or unauthorized user who can gain access an aggregate searchable encryption key. Hence, to overcome this problem, we use an identity-based encryption scheme to send the encrypted an aggregate searchable encryption key via mail.

1) Key generation: In our system, a master key S is the part of our setup algorithm. Each registered user U_i will have a public key P_i.

$P_i = H(ID_i)$ Where ID_i is the identity of the user U_i ... (8)

$S_i = S \times P_i$ Where S_i is the private key of the user U_i ... (9)

2) Encryption: Suppose, data owner wants to send an encrypted aggregate searchable encryption key to valid user through mail, she/he does the following:

- It picks a random number $r \in Z_p$ and then computes CA= $r * H(ID_d)$ where $H(ID_d)$ is the hash of data owner's email address which also is data owner's public key and generate the encryption key

$$Key_{AB} = e(S_d, H(ID_u))^r \text{ XOR } e(S_d, H(ID_u)) \quad \text{..... (10)}$$

Where S_d is Data owner's private key, $H(ID_u)$ is user's public key.

- After an encryption key Key_{AB} is generated, data owner passes the encryption key and original aggregate searchable key to the Advanced Encryption Standard (AES) algorithm [21], and this algorithm returns data owner an encrypted aggregate searchable key. After that data owner sends the encrypted aggregate searchable key along with C_A to user.

3) Decryption: When user receives the encrypted an aggregate searchable key from data owner, she/he does the following to get back the original aggregate searchable key.

- Generate decryption key

$$Key_{BA} = e(C_A, S_u)\ XOR\ e(H(ID_d), S_u) \ldots\ldots (11)$$

Where C_A is the multiplication of data owner's email address and the random number r, S_u is user's private key, and ID_d is data owner's public key. Upon user received the aggregate, she/he generate the decryption key Key_{BA} and then pass Key_{BA} along with an encrypted aggregate searchable key to the AES algorithm, and this algorithm returns the original aggregate searchable key back to user. Since, the Key_{AB} that data owner used to encrypt an aggregate searchable key actually equals to Key_{BA}, so user can use the derived Key_{BA} to decrypt the aggregate searchable key.

V. IMPLEMENTATION AND RESULTS

We have used Java Pairing-Based Cryptography (JPBC) library [12] for system parameter setup, and pairing calculation. We used Type-A pairing which is constructed on the curve $Y^2 = X^3 + X$ over the field F_p for some prime p.

1) Setup the system parameter

Cloud service provider uses setup algorithm to setup the scheme. Its take the input of parameter name and the maximum possible number n of files which belongs to a data owner, it outputs the public system parameter *params.*

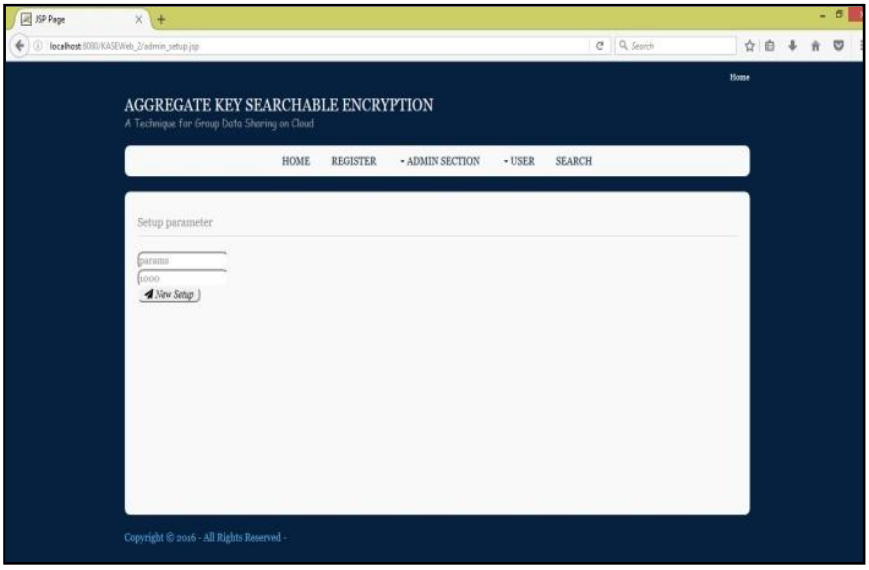

Figure 5.1 Setup parameter

2) Aggregate key generation by data owner

Before uploading file to the cloud, data owner must assign desire keywords for each file.

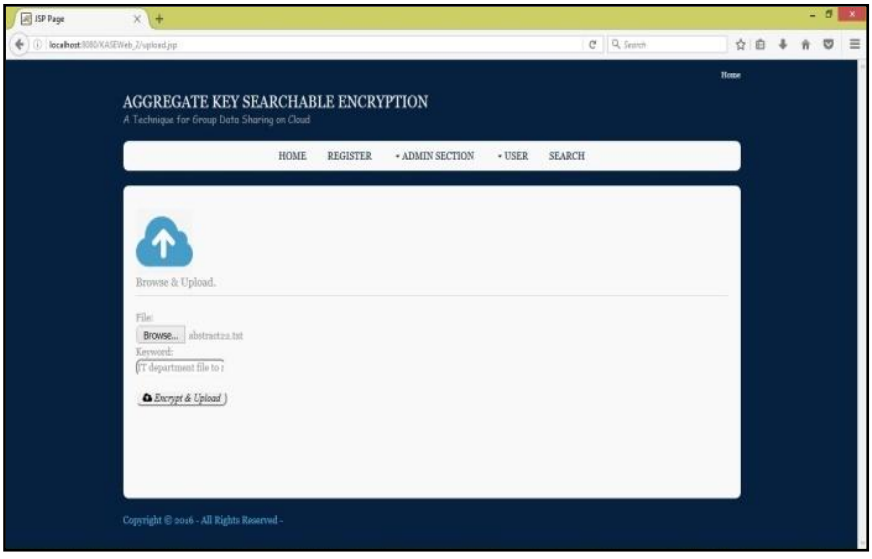

Figure 5.2 Encrypt and upload file to cloud storage

3) Aggregate key generation by data owner

Data owner generates a single aggregate for group of files to user for delegate keyword search rights for search over those encrypted files.

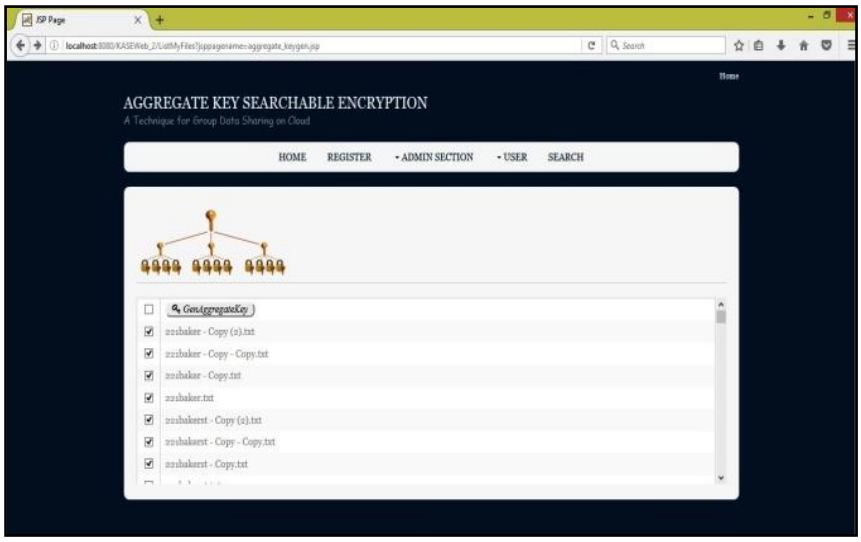

Figure 5.3 Aggregate key generation

*4) Authorized user used aggregate key to generate trapdoor***.**

To generate a trapdoor, Authorized user requires aggregate key as well as keywords of the particular file. Then generated aggregate trapdoor send to the cloud sever.

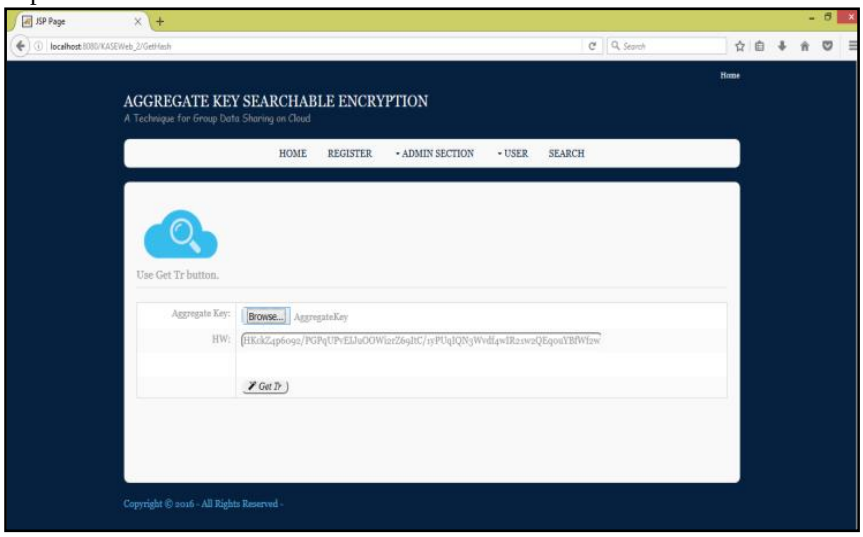

Figure 5.4 Trapdoor generation

5) Perform keywords search over shared data

After receiving trapdoor from authorized user, cloud server will adjust right trapdoor for each different file. Then, cloud server performs searches over encrypted data and retrieve matching keyword file.

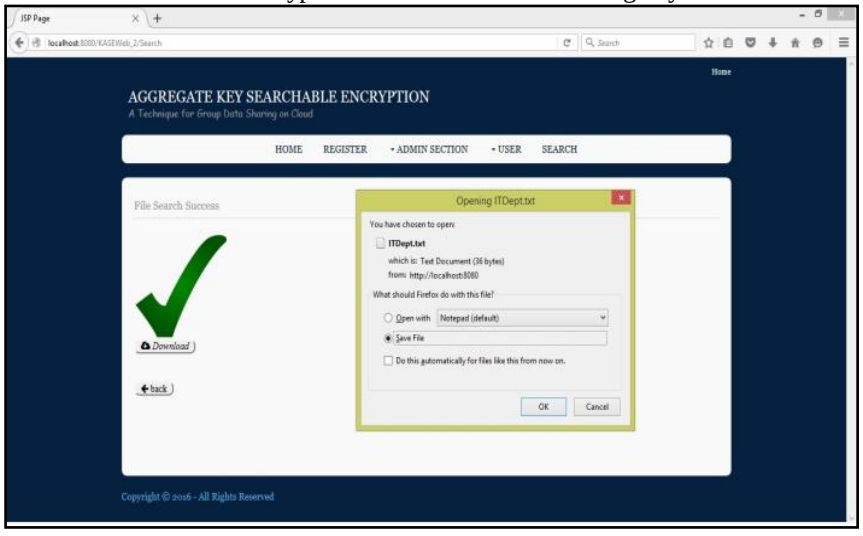

Figure 5.5 File search successfully

VI. PERFORMANCE EVALUATION

We evaluate the performance of our system based on Time cost. Time cost is the computation time actual algorithms take to run. Performance of our algorithms as follows:

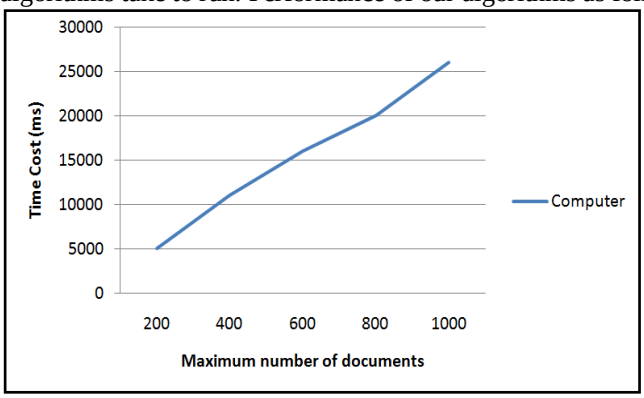

Figure 6.1 Time cost of Setup

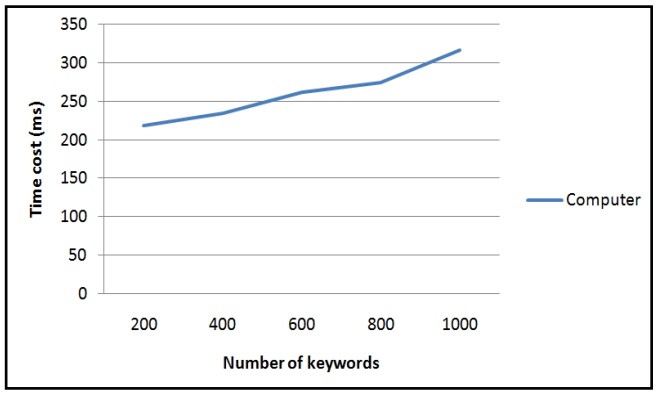

Figure 6.2 Time cost of Encrypt

As shown in the Figure 6.1, the execution time of Setup algorithm is linear in the maximum number of files belonging to one owner. When the maximum number of files grows up to 1000, it is required time for setup algorithm 26000 ms only. The execution time of encrypt algorithm is linear in the maximum number of keywords. When the maximum number of keywords grows up to 1000, it is required time for encrypt algorithm 316 ms only.

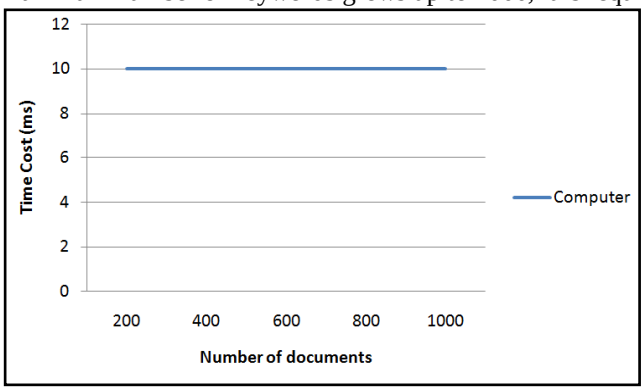

Figure 6.3 Time cost of Trapdoor

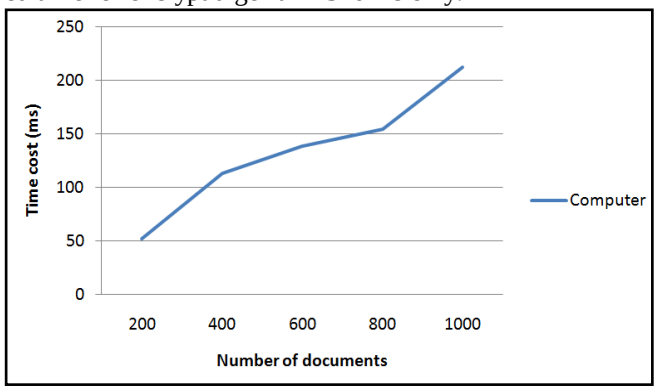

Figure 6.4 Time cost of Adjust

The execution time of Trapdoor is a constant. When the maximum number of files grows up to 1000, it is required time for trapdoor algorithm 10 ms only.The execution time of Adjust is linear in the number of files. The execution time for 1000 files is 212 ms only.

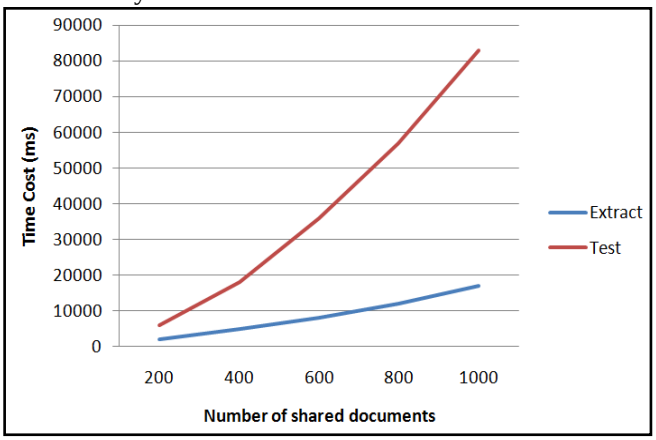

Figure 6.5 Time cost of Extract key and Test

As shown in the figure 6.5, the execution time of test algorithm is increases with the number of shared files. The execution time for extract algorithm for share 1000 files is 17000 ms only and the execution time of test algorithm for perform keywords search over 1000 files is 83000 ms only.

VII. CONCLUSION AND FUTURE WORK

In the traditional approach, data owner distributes multiple keys for sharing multiple files to the user and user submits multiple trapdoors to perform searches over multiple files. In our approach, data owner distributes single key for sharing multiple files to the user and user submits single trap door to perform searches over multiple files. We also

increase the security of an aggregate key by sending an encrypted form via mail. From the performance evaluation, we conclude that when number of shared files increases, searching time also increases.

In our scheme, data owner distribute a group of files with users and users submit a single trapdoor to perform searches over those files shared by same owner. Suppose, the user wants to search over those files shared by multiple data owners, then he must generate multiple trapdoors to the cloud. How to reduce the number of multiple trapdoors under multi-owner environment is a future work.

REFERENCES

[1] X. Song, D. Wagner, A. Perrig. "Practical techniques for searches on Encrypted data", IEEE Symposium on Security and Privacy, IEEE Press, pp. 44C55,2000.

[2] R.curtmola, J. Garay, S. Kamara, R.Ostrovsky. "Searchable symmetric encryption: improved definitions and efficient constructions", in: Proceedings of the 13th ACM conference on Computer and Communications Security, ACM Press, pp.

[3] D. Boneh, C. G, R. Ostrovsky, G. Persiano. "Public Key Encryption with Keyword Search", Eurocrypt2004, pp.506C522, 2004

[4] G J. W. Li, J. Li, X. F. Chen, et al. "Efficient Keyword Search over Encrypted Data with Fine-Grained Access Control in Hybrid Cloud", In: Network and System Security 2012, LNCS, pp. 490- 502, 2012.

[5] F. Zhao, F. Zhao, T. Nishide, K. Sakurai. Multi-User Keyword Search Scheme for Secure Data Sharing with Fine-Grained Access Control. Information Security and Cryptology, LNCS, pp. 406-418, 2012.

[6] R. A. Popa, N. Zeldovich. "Multi-key searchable encryption" cryptology ePrint Archive, Report 2013/508, 2013.

[7] C.Chu, S.Chow, W.Tzeng, et al. "Key-Aggregate Cryptosystem for Scalable Data Sharing in Cloud Storage", IEEE Transactions on Parallel and Distributed Systems, 2014.

[8] D. Boneh, C. Gentry, B. Waters. "Collusion resistant broadcast encryption with short Cipher-texts and private keys", Advances in Cryptology CCRYPTO 2005.

[9] C. Wang, Q. Wang, K. Ren, and W. Lou, "Privacy-Preserving Public Auditing for Data Storage Security in Cloud Computing", Proc. IEEEINFOCOM, pp. 525- 533, 2010.

[10] S. Yu, C. Wang, K. Ren, and W. Lou, "Achieving Secure, Scalable, and Fine-Grained Data Access Control in Cloud Computing", Proc. IEEE INFOCOM, pp. 534-542, 2010.

[11] NIST Privacy and Security guidelines (2012) NIST. Source: http://csrc.nist.gov/publications/nistpubs/800-144/SP800-144.pdf. Accessed on Oct 2012.

[12] B. Lynn, "The pairing-based cryptography library," http://crypto.stanford.edu/pbc/

[13] A. D. Caro and V. Iovino, "JPBC: Java Pairing Based Cryptography," 2011.

[14] P. Van,S. Sedghi, JM. Doumen. "Computationally efficient searchable symmetric encryption", Secure Data Management, pp. 87-100, 2010.

[15] X. Liu, Y. Zhang, B. Wang, and J. Yan. "Mona: secure multi owner data sharing for Dynamic groups in the cloud", IEEE Transactions on Parallel and Distributed Systems.

[16] M. Chase and S.S.M. Chow, "Improving Privacy and Security in Multi-Authority Attribute-Based Encryption," Proc. ACM Conf. Computer and Comm. Security, pp. 121-130. 2009

[17] D. Boneh and M.K. Franklin, "Identity-Based Encryption from the Weil Pairing," Proc. Advances in Cryptology (CRYPTO '01), vol. 2139, pp. 213-229, 2001.

[18] How PGP works" http://searchsecurity.techtarget.com/de_nition/Pretty-Good- Privacy, 1991.

[19] Changyu Dong "Math in Network Security: A Crash Course" www.doc.ic.ac.uk.

[20] Sattam S. Al-Riyami "Cryptographic Schemes based on Elliptic Curve Pairings"http://www.isg.rhul.ac.uk

[21] Avi Kak. AES: The Advanced Encryption Standard, Lecture Notes on Computer and Network Security, https://engineering.purdue.edu/kak/compsec

[22] V. Goyal, O. Pandey, A. Sahai, and B. Waters, "Attribute-Based Encryption for Fine-Grained Access Control of Encrypted Data,"Proc. 13th ACM Conf. Computer and Comm. Security (CCS '06),pp. 89-98, 2006

[23] Symmetric Encryption" http://cr.yp.to/bib/2004/bellare-chap4.pdf

[24] J. H. Silverman. "The arithmetic of elliptic curves." Springer-Verlag, Berlin, 1995

[25] American National Standards Institute ANSI X9.63. Public key cryptography for the financial services industry: Key agreement and key transport using elliptic curve cryptography," 2001.

[26] R. Sakai, K. Ohgishi, and M. Kasahara. Cryptosystems based on pairing. In The 2000 Symposium on Cryptography and Information Security" , Okinawa, Japan, 2000.

[27] A. K. Lenstra and E. R. Verheul. Selecting cryptographic key sizes." Journal of Cryptology, 2001.

[28] A. Shamir, Identity-based cryptosystems and signature schemes," Advances in Cryptology-Crypto'84, Lecture Notes in Computer Science, Vol.196, Springer-Verlag,pp. 47

