Efficient Keyword Based Search on Encrypted Cloud Data

Pooja Shah[#1], Gopal Pandey[*2]

[#1]*Department of Information Technology, Shantilal Shah Engineering College, Bhavnagar*
[1]poosrk12@gmial.com

[*] In-charge Head of the Department, Department of Information Technology,

Sir Bhavsinhji Polytechnic Institute, Bhavnagar

[2]*mr.gopal.pandey@gmail.com*

Abstract— **Cloud computing can be defined as a new style of Computing in which the resources are provided online through the internet. Cloud storage services allow the users to outsource their data in the cloud storage servers and retrieve them whenever and wherever required. This avoids the cost of building and maintaining their own data store. But the users need to provide privacy for the data and also needs to be able to search it without losing privacy. The users always search their documents through keyword in plaintext, which may leak privacy of users in cloud storage environment. So allowing a Cloud Service Provider (CSP), whose purpose is mainly for making a profit, to take the custody of sensitive data, raises underlying security and privacy issues. To keep user's data confidential against untrusted cloud service provider or outsiders, the normal way is to apply cryptographic primitives by giving the secret keys only to authorized users. In this paper we propose an efficient, secure and privacy preserving keyword search scheme which supports multiple users with low computation cost and flexible key management.**

Keywords— **Cloud Computing, Security, Searchable Encryption, Confidential Data, and privacy Preserving keyword search.**

I. INTRODUCTION

Cloud Computing is online method to distributing the resources on demand of users. And manages and schedules the computing resources through network. It constitutes a large computing resources pool which can provide service to users on their demand. Cloud Storage Services provides the large space of data storage resources and it stores the data on the remote servers based on the Cloud Computing. It uses the normal hard drive for storing the all kinds of the data on the remote servers. It includes database-like services and network attached storage. It is often billed for usage per gigabyte per month. For example, the Drop box simple storage service that provides 2GB free storage space and the after charges starting from $9.99 for 100, 200 and 500 GB per month.

The main purpose of outsourcing data to cloud is to reduce the computational overhead of the users. Consider a user has to store a document in cloud and later want to retrieve the files containing a particular keyword only. Normal practice is to store the document in the form of plain text which will introduce security and privacy risks. The cloud server is not fully trusted. Also there will be external attacks from unauthorized outsiders. To provide privacy, usual approach is to encrypt the data before it is stored in cloud. If a simple encryption scheme is used, the service provider could not determine which documents contains the specified keywords and could only return all the documents back to the user. Then the user will have to decrypt all the documents and get the required documents. This will depletes large computational power and too much memory of the client. The thin client will have only limited CPU power, memory and bandwidth. So a simple encryption scheme will not work well in this situation.[3][4]

To avoid this problem we can use the some natural approached like searchable encryption scheme. Now Qin liu introduced an privacy preserving keyword search scheme in cloud computing. That uses this searchable encryption scheme. In this approach we can use encrypted search keyword for finding the document. Now in this paper, we propose an efficient encrypted keyword search scheme suitable for cloud storage. It has the following advantages:

1) It supports keyword search in encrypted form. The cloud server could determine which all documents contain the specified keyword without knowing anything about the contents of document or the keyword searched.

2) The service provider will participate in the partial decipherment of the cipher text, thus reducing the computational overhead of the user.

3) Same keywords are encrypted to different cipher text for different documents thus reducing redundancy and avoiding the chance of statistical attack on keyword cipher text. So for that every time random key is generated.

II. RELATED WORK

It is an important research problem to enable the cloud service provider to efficiently search the keyword in encrypted form on encrypted files and providing user data privacy at the same time. Boneh et al introduced a public key encryption with keyword search (PEKS) which supports encrypted keyword search. Here the document is encrypted with any public key encryption algorithm and the user needs to decrypt it completely by him. It will depletes too much CPU and

memory power of the client if documents are decrypted frequently and loose the critical value of cloud computing.[3] Qin liu introduced an efficient privacy preserving keyword search which allows the service provider to participate in partial decipherment of the searched documents thus reducing the computational overhead of the user. But this scheme supports only single user. That is the user searching for data is same as the user who store data in cloud. In this scheme, the keyword encryption uses public key of all share users[4]. The trapdoor is made as the keyword query with the partial of every computed public key. When the number of users increases, it has low efficiency and it is not so efficient for the actual application of cloud. In this scheme also we used the symmetric key encryption. So it resists the statistical attack on keywords. Here same keywords are encrypted to not a same cipher text.

III. PROPOSED METHOD

The system consist of four different entities: data owner, data user, key server and the cloud server [5]:

Data Owner: Data owner has a collection of documents to be outsourced to the cloud in the encrypted form. Data owner will also determine which all users have access to the file and also distribute the secret key for authorization of the users.

Data Users: Data users are authorized persons who can retrieve the file stored in the cloud servers. They search for the files using trapdoor for keywords and can retrieve the file and decrypt it using the secret key.

Key Server: Key server is a trusted server which stores all the keys used for encrypting the document along with the signature of file names. The key server provides the key for decrypting a file to the user after verifying the signature of file name.

Cloud Server: Cloud server is an untrusted server which provides the storage service. The data owners store their documents along with encrypted keywords in the cloud storage server in encrypted form. Cloud server does not get any information about the document or the keywords.

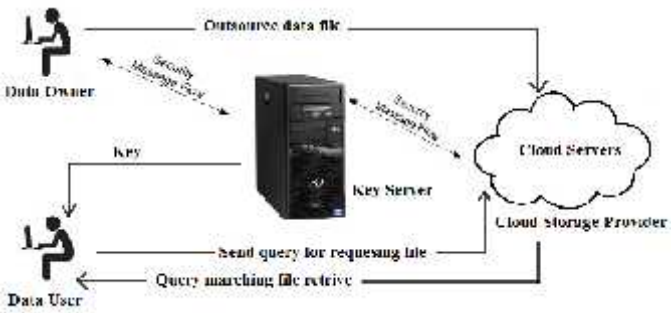

Figure 1 Secure Cloud Storage basic System Model

IV. ALGORITHM

This schema consists of the eleven randomize polynomial time algorithms as follows:

1. **KeyGen:** It takes sufficiently large security parameter K_1 as an input and produced the public/private key pair (Spub, Spriv) for the CSP. That we write KeyGen (K_1) = (Spub, Spriv).
2. **FEnc:** It is the semantic and public key encryption algorithm. It takes the owners secret key and CSP's public key and file F as input and generate the cipher text of the file known as C. We write FEnc (K_S, Spub, F) = C.
3. **KWHash:** It applies the hash function to the set of keywords like W = {W_1, W_2, ..., W_N} and generate hash values of the all keywords. After appending the time stamp t to each keyword, the hash function is again applied on this and generate cipher text of the keyword that known as $C_{Wi} \in C_W$.
4. **F_NHash:** The hash function is applied on the name of the file name which is used for searching the key for decryption of the cipher text of file that known as C_{NF}.
5. **R_WH:** It pickups the five random words from the file content having a word length greater than the threshold value and apply the hash function to this keyword that known as C_{RW} = {F_{W1}, F_{W2}, F_{W3}, F_{W4}, F_{W5}}.
6. **Fsign:** It generates the signature of the filename taking the value of cipher text of filename C_{NF}, secrete key K and C_{RW}.
7. **TCompute:** It computes the hash value of the keyword $W_I \in W$ and takes value of timestamp from CSP and again computes the hash value with append this to previously calculated hash value. That known as T_{WI}.
8. **KWTest:** It checks the document that contain requested keyword or not.
9. **PDecrypt:** It takes the Spriv Service private key and partially decrypts the document before sending to user.
10. **FTest:** When user gets the partially encrypted file and sends the request to key server for decrypting key of file then it sends the cipher text of file name. Key server runs this algorithm for searching the same file. If it matches then returns the cipher text filename with key and hash value of file random selected keywords. If it does not match then returns null.
11. **Recovery:** It takes the users secrete key and partially decrypted document as an input and generates the original document.

V. WORKING PROCESS

This schema working under the Semantic keyword based search. That uses the both semantic and public key encryption algorithm. The working process of this model follows:

Process 1: The owner generates secrete key by semantic algorithm each time when it store the document. The CSP generates the private/public key pair using public key encryption technique and stores its key to key server. This process is done by executing the ***KeyGen*** algorithm of our schema.

Process 2: Data Owner sends the following message to the Cloud Storage Server.

$MSG_{O2CSS} = [C, C_{Wi}, C_{NF}, C_{RW}, t]$,

Where *C* is Cipher text of the file, C_{Wi} is the Cipher text of the keywords, C_{NF} is the Cipher text of the filename, C_{RW} is the hash value of the random words from file content and ***t*** is the time stamp. That all terms are generated by executing the different algorithms that described below:

1. ***FEnc:*** The owner encrypts his document/file with secure semantic encryption algorithm *E* that generates the cipher text of file F with secrete key K that is randomly selected by owner. After that again it is encrypted with public key of Cloud Storage Server. So, we can write this $C = E_{Spub}(E_K(F))$.
2. ***KWHash:*** In our system, each file *F* is associated with a set of keywords $W = \{W_1, W_2, ..., W_N\}$. The owner applies the hash function on keywords $W_I^* = \{H(Wi)\}$ where (i=1, 2, 3… n). After appending the time stamp ***t*** to each keyword, the hash function is applied again to the encrypted hash value of the keywords $Cwi = H(t\ H(Wi))$ where i=1, 2,…. n and it returns the cipher text of the corresponding keywords $\{Cw_1, Cw_2, \ldots, Cw_n\}$.
3. ***FnEnc:*** The name of the File *F* is N_F, owner applies hash function H to compute $C_{NF} = H(N_F)$, C_{NF} which is used for searching for the key for decryption of the cipher text of *F*.
4. **RwH:** The random function picks up five random words from the file content having word length greater than a threshold value (in our case 5), then apply the hash function to the those keywords $C_{RW} = H(F_{W1}, F_{W2}, F_{W3}, F_{W4}, F_{W5})$.

If the above all function are executing than it generate the message that contain all values with adding time stamp *t* and sends to Cloud Server.

Process 3: After sending the information to CSS, The owner sends required information by following message to key server.

$MSG_{U2KS} = [C_{NF}, K, C_{RW}]$.

Here owner sends C_{NF}, the key K and the hash value of the selected random words from the file C_{RW} to the key Server. Now ***FSign*** function is executed and generates the file signature. In this encryption process, the user is not required to store any key, so addition or deletion of the user cannot affect this scheme. When adding users our scheme need not distribute any key, and when deleting user does not delete any key. For these reasons, our scheme reduces much of the computational overhead and makes key management an easy task.

Process 4: Now any data user U wants to search the keyword *Wm*, he firstly computes $TW_m = H(W_m)$, then sends *W*m to the cloud storage server. This value is generated by executing the ***TCompute*** algorithm. That is known as trapdoor. After computing trapdoor owner sends the message to database that is following:

$MSG_{U2CSS} = TWm = [H(W_1), H(W_2), .., H(W_n)]$

Process 5: After receiving *Wm*, the cloud storage server computes $Cwm' = H(t(H\ W_m))$. Here ***t*** is the timestamp of some file. If the cloud storage server finds a file whose *Cwm* is equal *Cwm'*. If match then the cloud storage server sends the corresponding Cipher text *C*, C_{NF}, *and* C_{RW} to user U, otherwise it returns NULL. This process is done by executing the ***KWTest*** algorithm. If the searching keyword matches more than one file, it results in many keyword matched files. To find the user desired file, it asks the user random word that is chosen from the file content, finds hash function for that word. It then matches this hash function with random words that are present in the already keyword matched files.

Process 6: If keyword of the file is match then the CSS perform the ***PDecrypt*** algorithm and corresponding Cipher text *C* is partially decrypted with the private key of CSS. It generate the intermediate output that known as C_P. It sends C_P, C_{NF} and C_{RW} to user U. For this the user has to know some words about the user desired file content information otherwise the user has to decrypt the all files relevant to the keyword match. The above process is represented by:

$MSG_{CSS2K'} = [C_P, C_{NF}, C_{RW}]$.

Process 7: user U sends C_{NF} to the key server, the key server compares C_{NF} with the information that is stored and if they are equal, returns the key *K* and C_{RW} to user U. This operation performs by executing ***FTest*** algorithm. In this process the following messages are passed between user and key server.

$MSG_{U2KS} = [C_{NF}]$, $MSG_{KS2U} = [C, K, C_{RW}]$.

Process 8: Then the user U needs to use *K* to decrypt C_P to get the file *F* and check the hash value of the random words C_{RW} for data integrity. If the Integration value C_{RW} differs, then the contents are changed by some malicious attacker otherwise content does not changed. This is done by using ***Recovery*** algorithm. So in this way we can provide the data integrity security and confidentiality because the authenticated users can access the information. The working process diagram of this proposed scheme is shown by **Figure 2.**

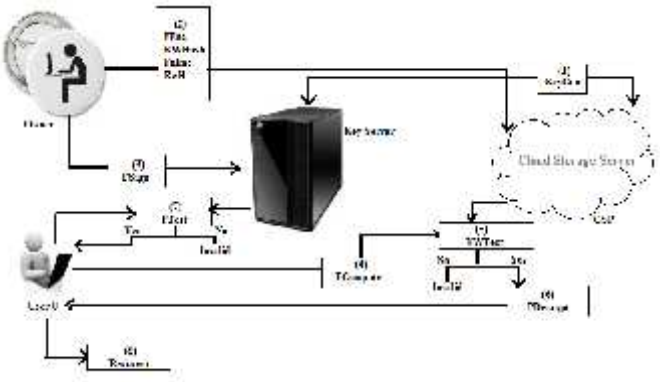

Figure 2 Process Diagram of Proposed Schema

VI. PERFORMANCE ANALYSIS

In The factors under consideration are key management, data redundancy, integrity and memory power & CPU capability in user's decryption for performance.

PEKS scheme uses the standard public key encryption algorithm for encrypting files without specifying a specific implementation. It requires all the decryption to be done by the user. This consumes a very long time for decryption. In EPPKS scheme the CSP participates in partial decryption to reduce the client's computational overhead, but it adds the burden to the server and also brings too much data redundancy in user querying.[5] The Table-I represents the performance comparison and analysis.

TABLE-1
PERFORMANCE COMPARISON

	PEKS	**EPPKS**	**Proposed Method**
Key Management	Hard	Hard	Easy
Data Redundancy	High	High	Less
Memory Power and CPU capability	High	Less	Very Less
Integrity	No	No	Yes

VII. CONCLUSION

In this paper, we have presented a new privacy preserving keyword based search scheme on cloud storage environment which reduces the time consumed in searching the data from the cloud storage system. It embargos the unauthorized users through the cipher text developed during the searching of data. It also provides solution to the data integrity problem raise during data transfer from cloud to the user and vice versa. In a symmetric cryptosystem this scheme proved semantically secure. It is also avoids statistical attack on keywords. In the proposed scheme, user authentication is provided before giving the secret key for decryption of document.

VIII. REFERENCES

[1] Seny Kamara, Kristin Lauter: Cryptographic Cloud Storage, Microsoft research.

[2] Tritty Mamachan1, Roshni. M. Thanka2: Survey on keyword searching in cloud storages , appear in International Journal of Emerging Technology and Advanced Engineering, ISSN 2250-2459, Volume 2, Issue 11, November 2012

[3] Liu Hong-xia, Dai Jia-zhu, Jiang Chao: Research on Privacy Preserving Keyword Search in Cloud Storage, appear in IEEE publication, 978-1-4244-5540-9/10, 2010.

[4] Qin Liuy, Guojun Wang, and Jie Wuz: An Efficient Privacy Preserving Keyword Search Scheme in Cloud Computing.

[5] Qin Liuy, Guojun Wang, and Jie Wuz: Secure and privacy preserving keyword searching for cloud storage services, appear in Journal of Network and Computer Applications, 9 March 2011.

[6] N. Cao, C. Wang, M. Li, K. Ren, and W. Lou, Privacy-Preserving Multi-Keyword Ranked Search over Encrypted Cloud Data,Proc. IEEE INFOCOM '11,2011