A Study of Key Compaction Encryption for Group File Sharing and Seeking Keyword in Cloud Storage

RAJASHRI[1],

M.Tech Scholar,
Department of Computer Science and Engineering
NMAM Institute of Technology, Nitte
574110,Udupi, India

Dr. SAROJADEVI HANDE[2]

Professor and Head,
Department of Computer Science and Engineering
NMAM Institute of Technology, Nitte
574110, Udupi, India

Abstract
Key compaction encryption for group data sharing and retrieval via cloud storage are used for effective management of encrypted data. They have flexibility of selectively sharing the data with multiple users in public cloud storage. Data owner can cluster many sets of secret keys and make them as compact as a single class. This approach provides formal security analysis of schemes in the standard model. Sharing group of selected files with any group of selected users requires different secret keys to be used for each document. However, there is an additional overhead for securely distributing each key to users, as those users will have to store equally large number of keyword cipher text in the cloud in order to perform a search over the encrypted data in cloud server. Large memory requirement for huge number of encrypted keys and computational complexity clearly renders this approach ineffective. To address this practical problem, a concept of Key Compaction Encryption in which a data owner needs to distribute a single aggregated key to the user for sharing huge number of documents has been utilized. The safety analysis and overall performance assessment performed confirmed that proposed scheme is comfortable for implementation. Later several concrete Key Encryption Schemes with distinct security tiers and extensions are tested with the proposed model.

Keywords: Public-Key Cryptosystem, Cloud data storage, Key Compaction Encryption, Keyword search.

Introduction
In recent times, sharing data with a group of users via cloud storage system has become very popular and has attracted much attention. Most of the data users share their personal files, such as photos, videos and business documents on daily basis with their friends using cloud storage system [1]. Even big business houses use cloud storage system due to various benefits associated with it such as lower cost and better resource utilization. But users are concerned with security of the data while sharing it in a cloud space due to risk of privacy and confidentiality [2]. To prevent data theft and exposure of data a common approach of encrypting the entire data file has been followed by the uploader, which can later be decrypted by the recipient using the secret keys. This methodology in cloud storage is called as cryptographic cloud storeroom [3, 4].These schemes are aimed to minimize the expense of storing the data and managing the secret keys for cryptographic use [5-8]. But the key granted to a branch of tree structure can be used to obtain keys for descendent nodes. Thus providing parent key can grant keys to all descendent nodes. Advanced cryptographic key assignment schemes are modeled with cyclic graph which support access policy [9-11]. These produce

keys for symmetric-key cryptosystems which can be very expensive.In existing systems unexpected privilege escalation will expose all documents. Also there is an unnecessary need for tracking all the documents for pattern search. This requires large number of keys to be disseminated to the users via protected channels for searching and decrypting of the files. Also, a large number of trapdoors need to be produced for keyword search which asks for large data storage space thus leading to computational complication which may cause the scheme to be ineffective. To overcome these demerits, a unique kind of public-key encryption concept called Key-Compaction Cryptosystem (KCE) has been proposed. In KCE, users encrypt a message underneath an identifier of cipher text referred to as class, which means that the cipher texts are similarly categorized into different classes. The key owner holds a master-secret key, which will be used to extract secret keys for distinct groups. The extracted key is combination of secret keys of all the documents in the group which is as compact as a single class i.e., it will be able to decrypt any subset of cipher text classes. The proposed scheme is illustrated in Fig. 1.1. Here owner simply delivers a single compacted key to end user via a secure communication media such as e-mail. End user can download the encrypted photos from owners drop box space and then use the single compact key to decrypt these encrypted photos by searching for photos by providing keyword text. Proposed novel concept of Key-Compaction Searchable Encryption (KCSE) algorithm can be applied to any cloud data storage system that supports keyword searchable group data sharing features, which means any person can selectively share a collection of files with a group of customers, and later allow them to carry out keyword search to seek the ciphered text documents.

To support searchable group file sharing and for economical key management measure needed are two-fold. First, data owner must send a single compact key to a data user for sharing large number of files instead of different keys to each file. Secondly the client only needs to submit a single trapdoor (instead of a group of trapdoors) to the cloud for performing keyword search for any number of shared encrypted files.In this work, safety analysis and overall performance assessment are performed on the proposed model. Later several concrete Key Encryption Schemes with distinct security tiers and extensions were tested on the developed model.

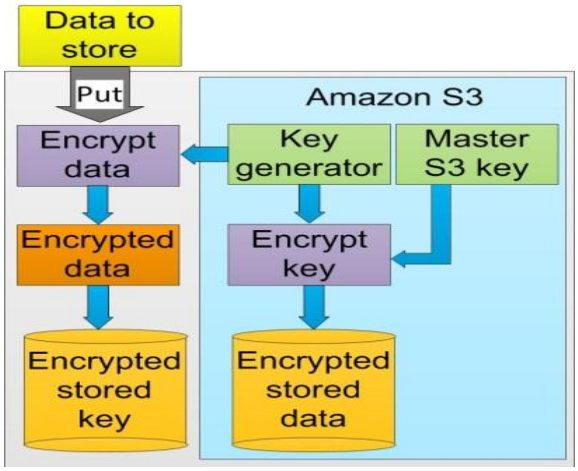

Fig. 1.1 Schematic of sharing documents with end user by sending a single compacted key.

The KCSE Framework

Preliminaries:

Assessment of few essential assumptions and cryptology ideas are needed in the framework of Key Compaction cryptosystem. Let M be the number of documents to be encoded and K be the compacted key for encoding and decrypting the text files and T_r the trapdoor for pattern search. The "Bilinear Diffie-Hellman (BDH)" hypothesis used extensively to prove the safety of a few "Broadcast Encryption (BE)"schemes. In a BE scheme, broadcaster encodes the documents set $M$for few sets of users S. Any data users in the set S can use the key to decrypt the encrypted message. A BEdesign is defined by three polynomial-set algorithms referred as "BE = (Setup, Encrypt, Decrypt)". "Searchable encryption (SE)" model has two operational method and is defined by three tuple "SE= (Setup, Encrypt, Trapdoor, Test)".

Framework:

In the structural module analysis of Key Compaction pattern, search scheme contains three procedural modules which are listed.

a) Cloud Server Module:

Cloud runs the following polynomial algorithm.

1. *Set Up (Params):* Cloud server establishes communication with the cloud service provider and the cloud client users by issuing the security parameters *Params*.
2. *Adjust(Params, i, S,*T_r): For producing the trapdoor for each of the document considering inputs are *Params*, *I*, *S*, T_r.
3. *Test keyword*(T_{ri},*i*): Used for checking the correct pattern in the encoded documents
.

b) Data Owner Module:

1. *Key Generation algorithm:* Is executed by the owner of the document to produce random key pair *pk* and *msk*.
2. *Encryption (pk, i):* For encryption, Advanced Encryption Standard(AES) and Data Encryption Standard(DES) algorithms are used. These algorithms are run to encode the i^{th} document and produce the cipher text data delta (Δ_i) for pattern searching keyC_i.
3. *Extract (msk, S):* This algorithm is used to produce a compacted encryption key for the delegates to decrypt and search for pattern in cloud. Output of this procedure is K_{agg}.

c) Data User Module:

1. *Trap Door (*K_{agg},*w):* This is used to perform search operation in the encoded data. Input of this algorithm is K_{agg} and pattern*w*. Output of this is trapdoor T_r.

Here,

n= maximum number of documents uploaded over cloud

S=Set of users in class S

i = Index of each document

pk= Public key

msk=Master Secrete Key

w = Keyword/pattern

K_{agg}= Aggregated Key

T_r = Trap door

Work Flow:

Implementation of Key Compaction Search Encryption pattern (KCSE) was done using Amazon S3 cloud storage (Simple Storage Service) system. Amazon web services are offered for file storage free of cost and additional charges are levied for premium service. System architecture is shown in fig1.2.

Step 1:System Setup

Here Key Compaction pattern search encryption scheme initially establish the communication between the data users and cloud server.

Step 2: User Registration

Data owner and data user register for the cloud server.

Step 3: User Login

This provides more security checks under the multi feature verification or digital signatures while both data user and data owners are logged into the cloud server. Typical examples are security parameters found in Drop Box and Citrix.

Step 4: File Uploading

Data owner uploads the file into amazon server in encoded form.

Step 5: File Sharing

Data owner can make a number of classes which he wishes to have in the cloud. For the file which he wishes to share with the delegates of different class, he can select the classes and send the aggregate key to all users of the selected class.

Step 6: Keyword Search

Data user can search the file over cloud by sending the pattern and can see the files sender is sharing with him.

Step 7: Data Downloading

Data user submits the received aggregated key and downloads the selected files which are later decrypted by using the same key.

Security Analysis:

In privacy-retaining record sharing, keyword seek is an essential requirement. Fortuitously, the Key-Aggregate Encryption (KAE) gives insights to the design of a KCSE scheme, even though the proposed scheme requires an extra complex mathematical transformation to help in keyword cipher text encryption and trapdoor generation. Here cloud server offers an effective way with pre-described schemes but, at the same time it may try to recover unwanted records based on its knowledge. Hence proving the authenticity of security features of the proposed scheme is very much essential.

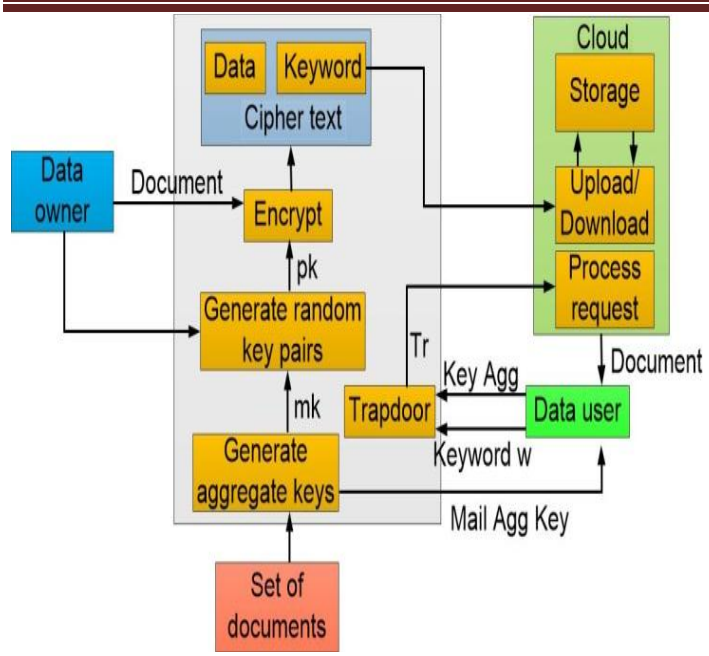

Fig.1.2 System architecture of key compaction algorithm.

Result Analysis

The safety analysis and overall performance assessment both confirm proposed schemes are probably comfortable. Several concrete Key Encryption schemes with distinct security tiers and extensions were tested within the proposed model.The periodical operational checking of the proposed system with the traditional encoding model shows the potential differences. File downloading and uploading was done without much delay in trend similar to the traditional method. Figure 1.3 and 1.4 shows faster accessing and retrieval of the data. Advantages of Key Compaction pattern search Encryption algorithm is that it supports data delegating methods in the cloud. Also there is no exposure of documents during patterned search of shared files. But Single key used for decryption and search should be sent via a secure transfer medium. Several concrete key encryption schemes with distinct security tiers and extensions tested were tested using the proposed model and were found to be satisfactory.

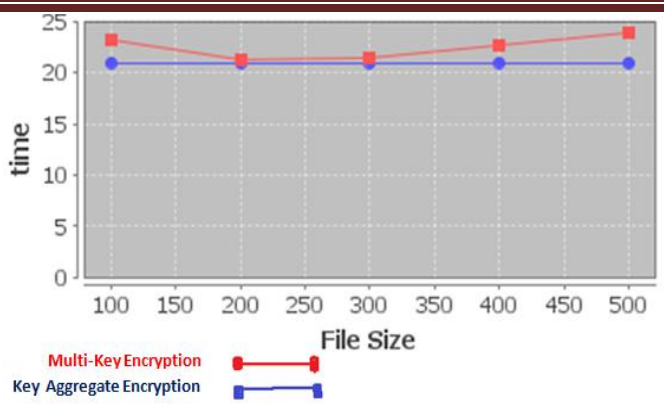

Fig.1.3 Performance comparison of aggregation time with key aggregation and multi-key encryption.

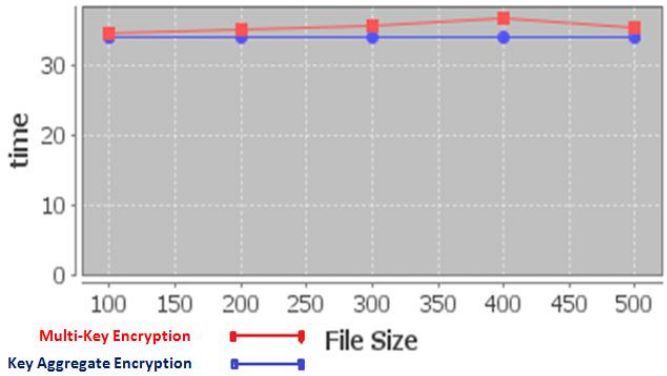

Fig.1.4 Performance comparison of encryption time with key aggregation and Multi-key Encryption.

Testing:

The proposed model was tested for its functionality. Table 1 provides the details of various test carried out. The functions were combined into different classes and the class as a whole was tested for its functionality. It was important to check for error-free interaction between various classes, and maintenance of data integrity.

Table 1 Integrated testing table

Class integrated	Functions integrated in each class	Tests done	Remarks
Class: Main	requestKey() createGroup() encryptFile()	Class tested to check whether all commands applied are working appropriately	Success
Class: Use Main	requestKey() DownloadFile() requestFile() generateTrapdoor() searchKeywordFromFile() requestAccessKey() decryptFile)	Class tested to check whether all commands applied are working appropriately	Success
Class: Key Generat-ion	generateKey()	Class tested to check whether all commands applied are working appropriately	Success

Testing validated the software's function in a manner that was reasonably expected out of a customer.

Table 2 Validation testing table

Class integrated	Functions integrated in each class	Tests done	Remarks
Working of front end	Users interaction with the help of mouse and keyboard	Appropriate forms open when clicked	Success
Working of data owner	Sender has to upload the file, encrypt the file and upload the encrypted file to amazon cloud	File upload to the cloud	Success
Working of Data user	End user has to download the file by first searching for the file using keyword then decrypting it	File downloaded from the cloud	Success

Conclusions

Need of the day is a sensible hassle free way of maintaining records privately and sharing them on a cloud storage system. Currently data owner needs to distribute large quantity of keys to customers to permit them to access his/her documents. This calls for an urgent need to have a simpler system of sharing keys which are secure but at the same time very efficient and theft proof. A concept of Key-compacted pattern Search Encryption (KCSE) was used in this work to these this hypothesis. Based on the analysis and evaluation of the proposed model it was observed that it can offer a powerful strategy to develop file sharing system based on public cloud storage. Using KCSE scheme, the proprietor can simply distribute a single key to a consumer while sharing plenty of documents and simply has to post a single trapdoor. Also if the customer desires to have a private way to access the data from multiple proprietors, he can ask the proprietor to generate more than trap door to the cloud. Since federated clouds have attracted a whole lot of attention these days, integrating KCSE with the federated clouds can be a scope for future study area.

References

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

[2]L.M. Kaufman, Data security in the world of cloudcomputing.IEEE Security & Privacy,Vol. 7, No. 4, pp. 61–64, 2009.

[3] S.S M. Chow, C.K. Chu, X. Huang, J. Zhou, and R. H. Deng, "Dynamic Secure Cloud Storage with Provenance," in Cryptography and Security: From Theory to Applications,Vol. 6805,pp. 442–464, 2012 .

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

Press, pp. 79-88, 2006.

[5] S.G. Akl and P.D. Taylor, "Cryptographic Solution to a Problemof Access Control in a Hierarchy," ACM Transactions on ComputerSystems (TOCS), vol. 1, No. 3, pp. 239–248, 1983.

[6] G.C. Chick and S.E. Tavares, "Flexible Access Control withMaster Keys," Proceedings of Advances in Cryptology – CRYPTO'89, Vol. 435, pp. 316-322, 1989.

[7] W.G. Tzeng, "A Time-Bound Cryptographic Key AssignmentScheme for Access Control in a Hierarchy," IEEE Transactions onKnowledge and Data Engineering (TKDE), vol. 14, No. 1, pp. 182-188,2002.

[8] G. Ateniese, A.D. Santis, A. L. Ferrara, and B. Masucci,"Provably-Secure Time-Bound Hierarchical Key AssignmentSchemes," Journal of Cryptology, Vol. 25, No. 2, pp. 243-270, 2012.

[9] Y. Sun and K.J.R. Liu, "Scalable Hierarchical Access Control inSecure Group Communications," Proceedings of the 23rdIEEEInternational Conference on Computer Communications (INFOCOM'04). IEEE, Vol.2, pp. 1296-1306, 2004.

[10] Q. Zhang and Y. Wang, "A Centralized Key Management Schemefor Hierarchical Access Control,"Proceedings of IEEE GlobalTelecommunications,Vol.4,pp. 2004.

[11] M.J. Atallah, M. Blanton, N. Fazio, and K.B. Frikken, "Dynamicand Efficient Key Management for Access Hierarchies," ACM Transactions on Information and System Security (TISSEC), Vol. 12, No. 3, 2009.