ISSN : 2347 - 8446 (Online)
ISSN : 2347 - 9817 (Print)
Vol. 4, Issue 2 (Apr. - Jun. 2016)
International Journal of Advanced Research in Computer Science & Technology (IJARCST 2016)

Identity Based Encryption Outsourced Revocation for Group Data Sharing Via Cloud

[I]Girish, [II]Subhash Kumar Gupta, [III]Dr. H. D. Phaneendra
[I]Associate Professor & Head, [II]PG Student CNE, [III]Professor
[I]Dept. of MCA, [II]Dept. of IS&E, [III]Dept. of CSE
[I,II,III]The National Institute of Engineering, Mysore, India

Abstract
The capability of selectively sharing encrypted data with different users via public cloud storage may greatly ease security concerns over inadvertent data leaks in the cloud. A key challenge to designing such encryption schemes lies in the efficient management of encryption keys. The desired flexibility of sharing any group of selected documents with any group of users demands different encryption keys to be used for different documents. However, this also implies the necessity of securely distributing to users a large number of keys for both encryption and search, and those users will have to securely store the received keys, and submit an equally large number of keyword trapdoors to the cloud in order to perform search over the shared data. The implied need for secure communication, storage, and complexity clearly renders the approach impractical. To share a file in a group of user's via cloud, In our proposed system we use Identity Based Encryption(IBE) for encryption as well as decryption of data. User who is sharing the file is called Data Owner and the other users who can access file is called Data User. Data owner only needs to distribute a single key to a user for sharing a documents, and the user only needs to submit a single trapdoor to the cloud for querying the shared documents. The security analysis and performance evaluation both confirm that our proposed schemes are provably secure and practically efficient.

Keywords
Cloud storage, use Identity Based Encryption(IBE), Data Owner , Data User

I. Introduction

Cloud storage has emerged as a promising solution for providing ubiquitous, convenient, and on-demand accesses to large amounts of data shared over the Internet. Today, millions of users are sharing personal data, such as photos and videos, with their friends through social network applications based on cloud storage on a daily basis.

IDENTITY-BASED Encryption(IBE) is an alternative to public key encryption,which is proposed to simplify key management in a certificate-based Public Key Infrastructure (PKI) by using human-intelligible identities (e.g., unique name, email address, IP address, etc) as public keys. There fore,sender using IBE does not need to lookup publickey and certificate, but directly encrypts message with receiver's iden- tity.Accordingly,receiver obtaining the private key associated with the corresponding identity from Private Key Generator (PKG) is able to decrypt such ciphertext. Though IBE allows an arbitrary string as the public key which is considered as an appealing advantages over PKI, it demands an efficient revocation mechanism. Specifically, if the private keys of some users get compromised, we must provide a mean to revoke such users from system. In PKI setting, revocation mechanism is realized by appending validity periods to certificates or using involved combinations of techniques .

II. Preliminary

Inthis section, we give abriefre view on identity based encryption

A. Identity-Based Encryption

An IBE scheme which typically involves two entities, PKG and users (including sender and receiver) is consisted of the following four algorithms.

- **Setup(λ) :** The setup algorithm takes as input a security parameter λ and outputs the public key *PK* and the master key *MK* . Note that the master key is kept secret at PKG
- **KeyGen(*MK,ID*) :** The private key generation algorithm is run by PKG, which takes as input the master key *MK* and user's identity ID□{0,1} . It returns a private key SK_{ID} corresponding to the identity .
- **Encrypt(*M,ID'*) :** The encryption algorithm is run by sender, which takes as input the receiver's identity ID' and a message *M* to be encrypted. It outputs the cipher- text *CT* .
- **Decrypt(CT,SK_{ID}) :** The decryption algorithm is run by receiver, which takes as input the ciphertext *CT* and his/her private key SK_{ID}. It returns a message *M* or an error .

An IBE scheme must satisfy the definition of consistency. Specifically, when the private key SK_{ID}generated by algo- rithm KeyGen when it is given *ID* as the input, then Decrypt(*CT*,SK_{ID}) where*CT*= **Encrypt**(*M,ID*) .

The motivation of IBE is to simplify certificate manage ment. For example, when Alice sends an email to Bob at bob@company com, she simply encrypts her message using Bob's email address "bob@ company com", but does not need to obtain Bob's public key certificate. When Bob receives the encrypted email he authenticate himself at PKG to obtain his private key, and read his email with such a private key.

III. Problem Statement

1. System Model

We present system model for outsourced revocable IBE in Fig.1. ComparedwiththatfortypicalIBEscheme,aKU-CSP is involved to realize revocation for compromised users. Actually, the KU-CSP can be envisioned as a public cloud run by a third party to deliver basic computing capabilities to PKG as standardized services over the network. Typically, KU-CSP is hosted away from either users or PKG, but provides a way to reduce PKG computation and storage cost by providing a flexible, even temporary extension to infrastructure. When revocation is triggered, instead of rerequesting private keys from PKG in [4],unrevoked users have to ask the KU-CSP for updating a lightweight component of their private keys. Though many details are involved in KU-CSP's deployment,

ISSN : 2347 - 8446 (Online)
ISSN : 2347 - 9817 (Print)

in this paper we just logically envision it as a computing service provider, and concern how to design secure scheme with an untrust KU-CSP.

Based on the system model proposed,we are able to define the outsourced revocable IBE scheme. Compared with the traditional IBE definition, the **KeyGen** ,**Encrypt** and **Decrypt** algorithms are redefined as follows to integrate time component. Note that two lists and are utilized in our definition, where records the identities of revoked users and is a linked list for past and current time period.

- KeyGen(MK,ID,RL,TL) :The keygeneration algorithm run by PKG takes as input a master key MK ,an identity ID ,a revocation list RL and a time list TL .If ID □ RL,the algorithm is aborted. Otherwise, it sends the private key SK_{ID} =(IK[ID],TK[ID]$_T$)to user where IK[ID] is the identity component for private key SK_{ID} and TK[ID]$_T$ is its time component for current time period T_i . Additionally, the algorithm sends an outsourcing key to KU-CSP.
- Encrypt(M,ID,T_i,PK) :The encryption algorithm run by sender takes as input–a message M , an identity ID and a time period T_i. It outputs the ciphertext CT .
- Decrypt(CT,SK_{ID}) : The decryption algorithm run by receiver takes as input–a ciphertext CT encrypted under identity ID and time period T_i and a private key $SK_{ID'}$=(IK[ID'],TK[ID']$_T$) .It outputs the original message M if ID=ID'and T_i=$T_{i'}$.

In addition, two algorithms are defined to realize revoca- tion at KU-CSP through updating the private keys of unrevoked users.

Revoke(RL,TL,{ID_{i1},ID_{i2}, ...ID_{ik}}) : The revocation algo- rithm run by PKG takes as input–a revocation list RL , a time listTL and the set of identities to be revoked{ID_{i1},ID_{i2}, ...ID_{ik}} . It outputs an updated time period T_{i+1} as well as the updated revocation list RL' and time list TL' .

KeyUpdate(RL,ID, T_{i+1} ,OK_{ID}) : The key update algorithm run by KU-CSP takes as input–a revocation list RL,an identity ID ,a time period, T_{i+1} and the outsourcing key OK_{ID} for identity . It outputs user's updated time component in private keyTK[ID] $_{Ti+1}$ if his identity does not belong to RL.

In this paper, we discuss user revocation, that is how to deprive users of decryptability even if they have been issued their private keys. To this end, we embed a time period into private key in a clever manner for revocation. Alice in our setting not only encrypts message with Bob's email address "bob@company. com" but also with current time period (e.g., "Thu Jul 18 2013"). When receives the encrypted email, Bob then obtains his private key consisting of an identity compo-nent and a time period component from PKG. With the both appropriate components, the email can be read. Suppose Bob is compromised. Then ,the time components of all the other users are updated by KU-CSP with a new time

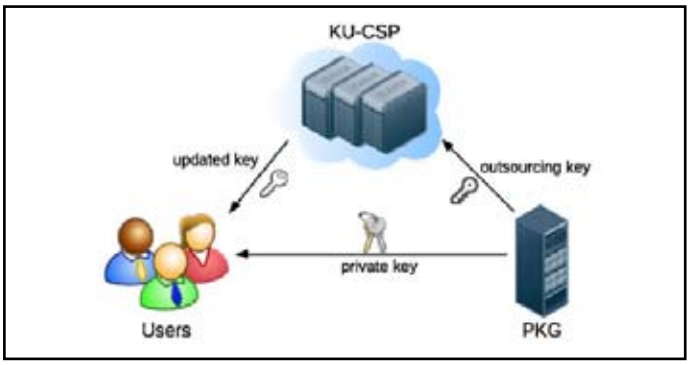


Fig. 1: System model for IBE with outsourced revocation.

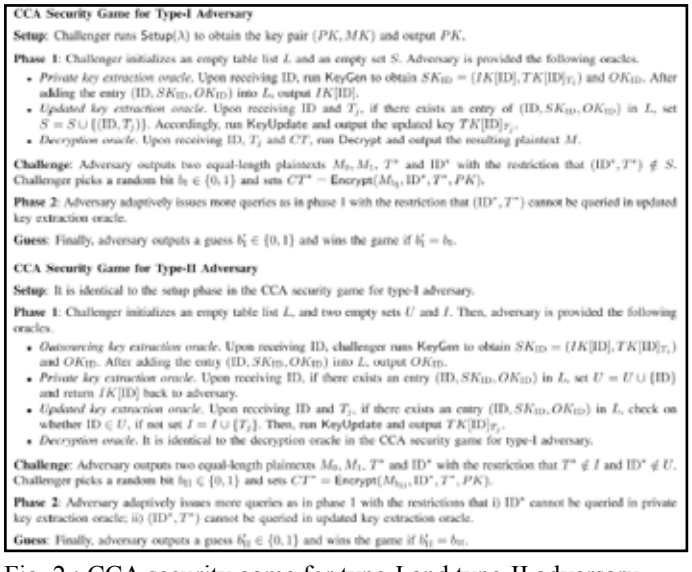
Period (e.g.,"FriJul192013").From then on ,the message sent to Bob should be encrypted with Bob's email address and the updated time period. Since Bob does not have the time component corresponding to the updated time period, the following encrypted messages cannot be decrypted by Bob even if they are intended for him. The challenge in designing the outsourced revocable IBE scheme is how to prevent a collusion between Bob and other unrevoked dishonest users. Specifically, a dishonest user (named Eve) can share her updated time component (i.e., "Fri Jul 19 2013") with Bob, and help Bob decrypt cipher text even if Bob just has the previous one (i.e., "Thu Jul 182013"). We will show how to avoid such a collusion later.

2. Security Definition

We assume that KU-CSP in the proposed system model is semi-trusted. Specifically ,it will follow our protocol but try to find out as much secret information as possible based on its possession. Therefore, two types of adversaries are to be considered as follows.

- *Type-I adversary* . It is de fined as a curious user with identity but revoked before time period T_i . Such adversary tries to obtain useful information from cipher text intended for him/her at or after T_i (e.g. time period T_i,T_{i+1}, . . .) through colluding with other users even if they are unrevoked. Therefore, it is allowed to ask for private key including identity component and updated time component for cooperative users. We specify that under the assumption that KU-CSP is semi trusted ,type-I adversary cannot get outsourcing key for any users.
- *Type-II adversary* .It is defined as a curious KU-CSP which aims to obtain useful information from cipher text intended for some target identity ID at time period T_i .Such adversary not only possess of outsourcing keys for all users in the system, but also is able to get user's private key through colluding with any other user with identity ID' . It is noted that to make such attack reasonable, we must restrict ID'≠ID.

Having the intuitions above, we are able to define CCA security game for type-I and type-II adversary respectively for our setting in Fig. 2. Suppose A_i is the type-i adversary for i=I, II . Then, its advantage in attacking the IBE with outsourced revocation scheme □ is defined as Adv □A(λ)=[Pr[b_i=b_i']-1/2]


CCA Security Game for Type-I Adversary

Setup: Challenger runs Setup(λ) to obtain the key pair (PK, MK) and output PK.

Phase 1: Challenger initializes an empty table list L and an empty set S. Adversary is provided the following oracles.

- *Private key extraction oracle*. Upon receiving ID, run KeyGen to obtain $SK_{\mathrm{ID}} = (IK[\mathrm{ID}], TK[\mathrm{ID}]_{T_i})$ and OK_{ID}. After adding the entry $(\mathrm{ID}, SK_{\mathrm{ID}}, OK_{\mathrm{ID}})$ into L, output $IK[\mathrm{ID}]$.
- *Updated key extraction oracle*. Upon receiving ID and T_j, if there exists an entry of $(\mathrm{ID}, SK_{\mathrm{ID}}, OK_{\mathrm{ID}})$ in L, set $S = S \cup \{(\mathrm{ID}, T_j)\}$. Accordingly, run KeyUpdate and output the updated key $TK[\mathrm{ID}]_{T_j}$.
- *Decryption oracle*. Upon receiving ID, T_j and CT, run Decrypt and output the resulting plaintext M.

Challenge: Adversary outputs two equal-length plaintexts M_0, M_1, T^* and ID* with the restriction that $(\mathrm{ID}^*, T^*) \notin S$. Challenger picks a random bit $b_1 \in \{0,1\}$ and sets $CT^* = \mathrm{Encrypt}(M_{b_1}, \mathrm{ID}^*, T^*, PK)$.

Phase 2: Adversary adaptively issues more queries as in phase 1 with the restriction that (ID^*, T^*) cannot be queried in updated key extraction oracle.

Guess: Finally, adversary outputs a guess $b_1' \in \{0,1\}$ and wins the game if $b_1' = b_1$.

CCA Security Game for Type-II Adversary

Setup: It is identical to the setup phase in the CCA security game for type-I adversary.

Phase 1: Challenger initializes an empty table list L, and two empty sets U and I. Then, adversary is provided the following oracles.

- *Outsourcing key extraction oracle*. Upon receiving ID, challenger runs KeyGen to obtain $SK_{\mathrm{ID}} = (IK[\mathrm{ID}], TK[\mathrm{ID}]_{T_i})$ and OK_{ID}. After adding the entry $(\mathrm{ID}, SK_{\mathrm{ID}}, OK_{\mathrm{ID}})$ into L, output OK_{ID}.
- *Private key extraction oracle*. Upon receiving ID, if there exists an entry $(\mathrm{ID}, SK_{\mathrm{ID}}, OK_{\mathrm{ID}})$ in L, set $U = U \cup \{\mathrm{ID}\}$ and return $IK[\mathrm{ID}]$ back to adversary.
- *Updated key extraction oracle*. Upon receiving ID and T_j, if there exists an entry $(\mathrm{ID}, SK_{\mathrm{ID}}, OK_{\mathrm{ID}})$ in L, check on whether $\mathrm{ID} \in U$, if not set $I = I \cup \{T_j\}$. Then, run KeyUpdate and output $TK[\mathrm{ID}]_{T_j}$.
- *Decryption oracle*. It is identical to the decryption oracle in the CCA security game for type-I adversary.

Challenge: Adversary outputs two equal-length plaintexts M_0, M_1, T^* and ID* with the restriction that $T^* \notin I$ and $\mathrm{ID}^* \notin U$. Challenger picks a random bit $b_{\mathrm{II}} \in \{0,1\}$ and sets $CT^* = \mathrm{Encrypt}(M_{b_{\mathrm{II}}}, \mathrm{ID}^*, T^*, PK)$.

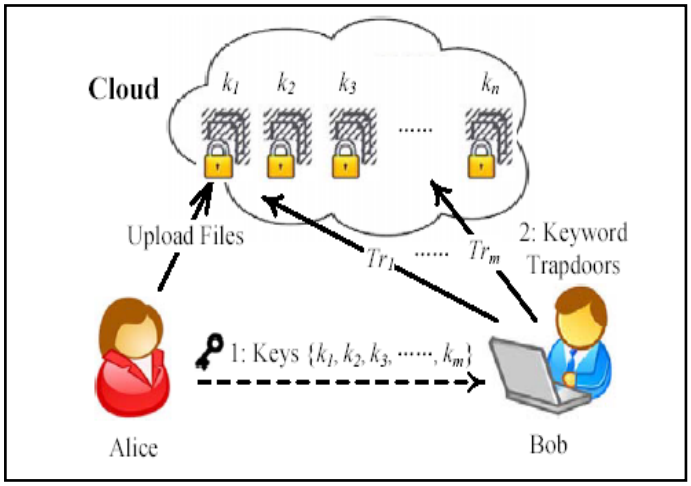
Phase 2: Adversary adaptively issues more queries as in phase 1 with the restrictions that i) ID* cannot be queried in private key extraction oracle; ii) (ID^*, T^*) cannot be queried in updated key extraction oracle.

Guess: Finally, adversary outputs a guess $b_{\mathrm{II}}' \in \{0,1\}$ and wins the game if $b_{\mathrm{II}}' = b_{\mathrm{II}}$.

Fig. 2 : CCA security game for type-I and type-II adversary.

3. Scenario

Consider a scenario where two employees of a company would like to share some confidential business data using a public cloud storage service (e.g., dropbox or syncplicity). For instance, Alice wants to upload a large collection of financial documents to the cloud storage, which are meant for the directors of different departments to review. Suppose those documents contain highly sensitive information that should only be accessed by authorised users, and Bob is one of the directors and is thus authorized to view documents related to his department. Due to concerns about potential data leakage in the cloud, Alice encrypts these documents with different



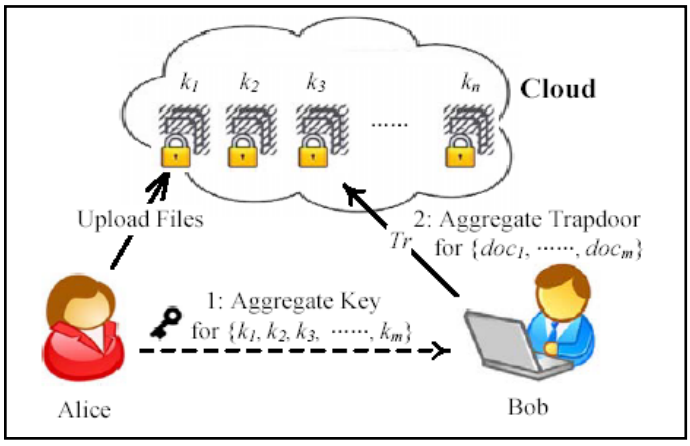
(a) Traditional approach

Fig. 3: keyword search in group data sharing system.

keys, and generates keyword ciphertexts based on depart- ment names, before uploading to the cloud storage. Alice then uploads and shares those documents with the directors using the sharing functionality of the cloud storage. In order for Bob to view the documents related to his department, Alice must delegate to Bob the rights both for keyword search over those documents, and for decryption of documents related to Bob's department.

With a traditional approach, Alice 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 these keys in order to perform a keyword search. As shown in Fig.1(a), Alice is assumed to have a private document set $\{doc_i\}_{i=1}^{n}$, and for each document doc_i, a searchable encryption key k_i is used. Without loss of generality, we suppose Alice wants to share m documents $\{doc_i\}_{i=1}^{m}$ with Bob. In this case, Alice must send all the searchable encryption keys $\{k_i\}_{i=1}^{m}$ to Bob. Then, when Bob wants to retrieve documents containing a keyword w, he must generate keyword trapdoor Tr_i for each document doc_i with key k_i and submit all the trapdoors $\{Tri\}_{i=1}^{m}$ to the cloud server. When *m* is sufficiently large, the key distribution and storage as well as the trapdoor gener- ation may become too expensive for Bob's client-side device, which basically defies the purpose of using cloud storage.

In this paper, we propose the novel approach of key-aggregate searchable encryption (KASE) as a better solution, as depicted in Fig.1(b). , in KASE, Alice only needs to distribute a single aggregate key, instead of $\{k_i\}_{i=1}^{m}$ for sharing m documents with Bob, and Bob only needs to submit a single aggregate trapdoor, instead of $\{Tr_i\}_{i=1}^{m}$, 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 Bob. Therefore, in KASE, the delegation of keyword search right can be achieved by sharing the single aggregate key. We note that the delegation of decryption rights can be achieved using the keyaggregate encryption approach recently proposed in [6], but it remains an open problem to delegate the keyword search rights together with the decryption rights, which is the subject topic of this paper.



(b) Key-Aggregate Searchable Encryption

IV. The Proposed Scheme

A. Overview

The design of our KASE scheme draws its insights from both the multi key searchable encryption scheme [6] and the key-aggregate data sharing scheme [7]. Specifically, in order to create an aggregate searchable encryption key instead of many independent keys, we adapt the idea presented in [6]. Each searchable encryption key is associated with a particular index of document, and the aggregate key is created by embedding the owner's master secret key into the product of public keys associated with the documents. In order to implement keyword search over different documents using the aggregate trapdoor, we employ a similar process as in [7]. The cloud server can use this process to produce an adjusted trapdoor for every document.

B. Efficiency

In terms of efficiency, our scheme clearly achieves constant-size key word ciphertext,trapdoor and aggregate keys. In addition, we should point out that:

1) The set S, which includes the indices of shared documents, has a linear size in the number of documents associated with the aggregate key. However, this does not affect the usefulness of the data sharing system, because the content of S can be safely stored in the cloud, such that there is no need to submit them to the cloud server when submitting the trapdoor.

2) The public system parameters PubK is O(n) in size, which is linear in the maximum possible number of documents belonging to a data owner, but not dependent on the number of documents stored in the cloud server, and hence this will not affect the system's practicality.

C. Security Analysis

To analyze the security of our scheme, and in particular show that the scheme satisfies the security requirements, we assume that the public cloud is "honest-but-curious". That is, the cloud server will only provide legitimate services according to predefined schemes, although it may try to recover secret information based on its

ISSN : 2347 - 8446 (Online)
ISSN : 2347 - 9817 (Print)

knowledge.

D. A Group Data Sharing System

When constructing a practical group data sharing system, it is important to reduce the number of keys belonging to a user. In this subsection, we will introduce how to build such a system based on the KASE and KAE schemes with the same public parameters.
We consider a group data sharing system without using any private cloud, but instead based on widely available public cloud services, such as Dropbox or citrix. Based on such a consideration, we assume a group manager (e.g., the HR director of an organization) with an authorized account to act in the role of "manager" who will be responsible for management of the system including maintaining the public system parameters stored in the cloud.

V. Conclusion

Considering the practical problem of privacy preserving data sharing system based on public cloud storage which requires a data owner to distribute a large number of keys to users to enable them to access his/her documents, we for the first time propose the concept of key-aggregate searchable encryption (KASE) and construct a concrete KASE scheme. Both analysis and evaluation results confirm that our work can provide an effective solution to building practical data sharing system based on public cloud storage.
In a KASE scheme, the owner only needs to distribute a single key to a user when sharing lots of documents with the user, and the user only needs to submit a single trapdoor when he queries over all documents shared by the same owner. However, if a user wants to query over documents shared by multiple owners, he must generate multiple trapdoors to the cloud. How to reduce the number of trapdoors under multi-owners setting is a future work. Moreover, federated clouds have attracted a lot of attention nowadays, but our KASE cannot be applied in this case directly. It is also a future work to provide the solution for KASE in the case of federated clouds

References

[1] *W.Aiello,S.Lodha,andR.Ostrovsky,"Fast digital identity revocation,"in Advancesin Cryptology(CRYPTO'98).New York,NY,USA: Springer, 1998, pp. 137–152.*
[2] *V.Goyal,"Certificate revocation using fine grained certificate space partitioning,"in Financial Cryptography and Data Security,S.Dietrich and R. Dhamija, Eds. Berlin, Germany: Springer, 2007, vol. 4886, pp. 247–259.*
[3] *F. Elwailly, C. Gentry, and Z. Ramzan, "Quasimodo: Efficient certificate validation and revocation," in Public Key Cryptography (PKC'04), F. Bao, R. Deng, and J. Zhou, Eds. Berlin, Germany: Springer, 2004, vol. 2947, pp. 375–388.*
[4] *D.BonehandM.Franklin,"Identity-based encryption from the Weil pairing," in Advances in Cryptology (CRYPTO '01), J. Kilian, Ed. Berlin, Germany: Springer, 2001, vol. 2139, pp. 213–229.*
[5] *A. Boldyreva, V. Goyal, and V. Kumar, "Identity-based encryption with efficient revocation,"in Proc.15th ACM Conf. Comput.Commun. Security (CCS'08), 2008, pp. 417–426.*
[6] *C.Chu,S.Chow,W.Tzeng,etal."Key-Aggregate Cryptosystem for Scalable Data Sharing in Cloud Storage", IEEE Transactions on Parallel and Distributed Systems, 2014, 25(2): 468-477.*
[7] *R. A. Popa ,N. Zeldovich. "Multi-key searchable encryption". Cryptology ePrint Archive, Report 2013/508, 2013.*