ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

Capability of Encryption in Cloud through KAC

Vajrala Krupa Rani
PG Scholar,
Dept of Computer Science & Engineering,
SMITW, Tummalapalem,
Guntur (Dt), A.P, India.

Pulipati Swaroop
Assistant Professor,
Dept of Computer Science & Engineering,
SMITW, Tummalapalem,
Guntur (Dt), A.P, India.

P.G.K Sirisha
Associate Professor & HOD,
Dept of Computer Science & Engineering,
SMITW, Tummalapalem,
Guntur (Dt), A.P, India.

Abstract:
Cloud storage is an important function in data sharing. And it is very secure, efficient and flexible compared to other software's. Here it can produce constant-size encrypted-texts by the description of new public-key cryptosystems (one type of algorithm used for secure), such that systematic duplication of description rights for any set of cipher texts are possible. Here one can cluster any set of unpublished keys and make them as a single key by the originality(novelty).But enclosing the power of all the keys existence collected. In other form the secret key holders can open a constant-size cluster key for adjustable option of cipher text set in cloud storage, but the other encoded files outside the set remain private. This concerted cluster key can change and sent to others or be stored in smart card with very restricted secure storage. We provide formal security analysis of our schemas in the standard model. We also derive another application of our schemes. In important our scheme provides the first public key patient controlled encryption for flexible hierarchy, which was identified.

1. INTRODUCTION:

Cloud storage is going rapidly in enterprise. It has demand for data outsourcing which accesses in the corporate data. It is also used in many online services for personal applicants, it is easy to create an account for email, photo albums and file sharing with storage space more than 25GB.By using wireless technology we can access emails, stored photos by a mobile phones in any corner of the world. Considering data privacy, a traditional way to ensure it is to rely on the server to enforce the access control after authentication.

Suddenly the authorized encapsulations will interpret the all data. Where the shared-tendency cloud computing environment will be there, in that things become placid not so good. Data will be reside in single physical machine, but the dada from various clients can be receive on separate Virtual Machine (VM).Data in objective VM could be sacked by instantiating separate VM co-resident with the host one[2].in relation to chance of files ,there are a consecution of cryptographic structure which spirit as a good way as acknowledging a third-party accountant to check the chance of files on favor of the data builders without divulging anything about the dada[3], or without adjusting the data builders inconspicuousness[4]. Similarly, cloud users apparently will not clasp the able acceptance that the cloud server is doing proper job in other words acquaintance.

A solution of cryptographic e.g.,[5],whenever the client is not confidently satisfy with trusting the safety of the VM or the bluntness of the technical staff, with ascertain security confide on number-theoretic assuming is more attractive. These users are instigated to encrypt their data along their own keys earlier uploading to server. The considerable functionality of cloud computing is data sharing for example if soul let their friends see a child set of their unpublicized pictures .Here how we will strong distributed encrypted data is the declarative problem of course from the storage clients can download encrypted data., decrypted them, then send them to various clients for sharing., but here it will lose their cloud storage. User's access deputes data from the server, because clients should be able to depute the access rights of

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

generosity data to another person. Here in cloud storage is not niggling, because however we finding an efficient and secure way to unpublished partial. Here example for illustration is Drop box. Assume the Dropbox having the photos of Alice, she don't want to aerate her photos on publish. Due to various data desolate possibilities; she

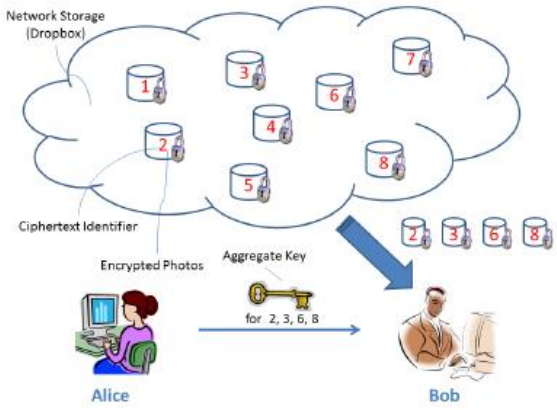


Fig:1 Alice shares files with Bob with single aggregate key

Doesn't receive her photos privacy with production mechanism improvised by Dropbox, so encrypt her photos with their own keys before uploading her photos. One day Alice friend Bob ask her photos to take over all these year which Bob glisten in. Now Alice can use the share function of Dropbox for sharing photos, but now the problem is how to depute the decryption rights to Alice photos to Bob. Now Alice assume to select appropriate option for sending photos to Bob is secure is involving secure keys to send of course there are two ways to encryption paradigm.

->Alice send encrypted file with single secure key, and Bob gives the directly by related secrete key.
->Alice use different keys for sending encrypted file and Bob also send related secrete keys.

Apparently, the first one is unsatisfied, since may be all un-select data is public to Bob. And the second one is partially concentrated on efficiency. And here many number of shared keys for shared photos will be there, these encrypted keys intrinsic require a encrypted channel, and strong these keys requires instead expensive encrypted storage. Here increasing cost, complexity because here we are maintain separate keys for each encrypted file in over words simply it having heavy wait and costly. The encrypted keys are mainly two types they are symmetric key and asymmetric key. When Alice organized her data from third party, the she use Symmetric encrypted data, apparently this is not coveted. The encrypted key and decrypted key different in public-key encryption by framing. The application can got more flexibility by the use of public-keys. For example every employee can upload their encrypted files on cloud storage server without the knowledge of company master secretes key.

2. KEY-AGGREGATE ENCRYPTION:

First of all we give what is framework and definition for key-aggregate encryption. After we discuss how to access KAC in a sequence of its application in cloud storage.

2.1 Framework:

A key-aggregate encryption system contains 5 polynomial-time algorithms as like. The information owner finds out the public scheme argument through Setup and produces a public/master-secret key pair through KeyGen. Data can be packed via Encrypt by somebody who also contexts what ciphertext category is combined with the plaintext data to be encrypted. The information owner can access the master-secret to produce an aggregate decryption key for a set of ciphertext categories Through Extract. The produced keys are passed to delegates securely (via e-mails or secure modules).Finally, any user with an aggregate key can decrypt any ciphertext produced that the ciphertext's category is maintained in the aggregate key through Decrypt. .Build$(1^{\lambda}, n)$:executed by the information owner to build an account on an untrusted server.On input a security level argument 1λ and the number of ciphertext categories n (i.e.,class index should be an integer bounded by 1 and n),it outputs the

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

public schedule argument param, which is removed from the input of the other algorithms for brevity.

Key Gen: executed by the information owner to randomly produce a public/master-secret key pair (pk, msk).

- Encrypt (pk, i,m):executed by someone who wants to packed information. On input a public-key pk, an index i representing the ciphertext category, and a data m, it outputs a ciphertext c.
- Extract (msk, S): executed by the information owner for delegating the decrypting power for a specific set of ciphertext categories to a delegate. On input the master-secret key msk and a set S of indices respective to dissimilar categories, it outputs the aggregate key for set S represented by Ks.
- Decrypt (K_S, S, i , C):executed by a delegate who accepted an aggregate key Ks produced by Extract. On input Ks, the set S, an index i representing the ciphertext class the ciphertext C related to , and C,it outputs the decrypted result m if I e S.

There are 2 functional requirements:

- Correctness For any integersλ andn,anysetS$\subseteq${1,⋯,n},anyindex i∈S and any message m, Pr[Decrypt(K_S,S,i,C) = m : param← Setup(1^λ,n), (pk,msk) ←KeyGen(),C ← Encrypt(pk,i,m), K_S ← Extract(msk,S)] = 1.
- Compactness For any integers λ, n, any set S, any index i ∈ S and any data m; param← build((1^λ,n), (pk,msk) ←KeyGen(),K_S ← Extract(msk,S) and C ← Encrypt(pk,i,m); |K_S| and |C| only depend on the security argument λ but individual of the number of categories n.

2.2 Sharing Encrypted Information:

A canonical application of KAC is information sharing. The key aggregation characteristic is particularly useful when we want the delegation to be flexible and efficient. The schedules unseen a data producer to distribute her information in a secureand elective path, with a small and fixed ciphertext expansion, by sharing to every authorized user a small and single aggregate key. Here we discuss the primary thought of information distributed in cloud storage using KAC, shown inFigure2.For example Alice wants to distribute her data s1,s2,...,svon the server. She first calculate build((1^λ,n) to get param and execute KeyGen to get the public/master-secret key pair (pk,msk).

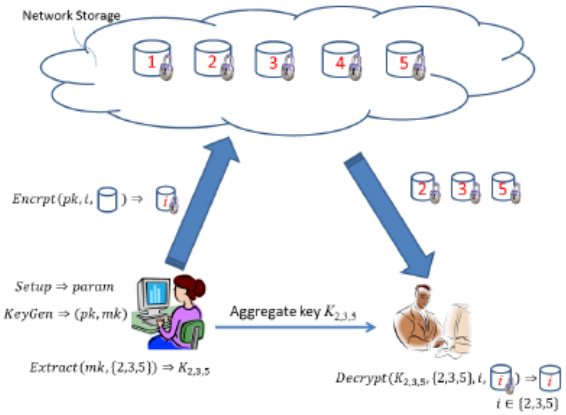


Fig:2 Using KAC for data sharing Cloud

The system argument param and public-key pk can be made public and master-secret key msk should be place secret by Alice. someone (including Alice herself) can then encrypt each mi by C_i = Encrypt(pk,i,mi). The encrypted information are updated to the server. With param and pk, public who coordinate with Alice can upload Alice's information on the server. Once Alice is ready to share a set S of her information with a friend Dob, she can calculate the aggregate key KS for dob by performing Extract(msk, S). Since KS is now a fixed size key, it is easy to be sent to Dob through a secure email. After obtaining the aggregate key, Dob can download the information he is authorized to access. That is, for every i ∈ S, Dob downloads C_i (and some required values in param) from the server. With the aggregate key K_S, Dob can decrypt everyC_i by Decrypt(K_S,S,i,Ci) for every i ∈ S

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

3. RELATED WORK:

In this session has we can compare our inferior KAC plan with other possible solutions on sharing in unpublished cloud storage. We can optimize our differences in Table 1.

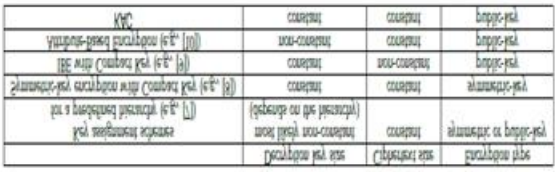


Table 1: Comparison between KAC and other Schemes

3.1 Cryptographic key for a Predefined Hierarchy'

Here we can begin by the exanimation the most related research in the literature of cryptography/security. Cryptographic key is a value to it a binding which will be visible scheme program outside of that blocks scope decrees the money in storing an controlling secrete keys for general cryptographic use. Using a tree diagram a key for a given node can be used to explain the keys of its big too small. Just permitting the super key implements permissions all the keys of its big to small nodes. Sandhu [15] a method to generate a tree hierarchy of symmetric keys by using more than one time estimation of pseudorandom functions/ block-cipher on a permanent secret. The concept can be generalized from a tree to a graph. More advanced pictographic is a value to its which will be a binding which will be visible scheme program outside of block code to promote policy that can be changed by an acyclic graph or a cyclic graph [16],[17],[7] most of the block outside produce keys for symmetric keys cryptosystem's, even though key equation may require modular arithmetic as used in public key cryptosystem, which are generally more costly than "symmetric key operation" such as pseudorandom function. We take the tree structure as an example. Alice can first classify the cipher-text classes according to their subjects like fig three. Each node in the tree represent secrete key, while the external nodes represents the keys for individual cipher-text classes.

All the nodes are invited to fill the circles represents the keys for the classes to be delivering by the persons and circles curcuma vented by dotted lines represents the key to be accepted. Note that every key of the non-leaf node can equate the keys of its big to small nodes. In fig-3(a) if Alice want to distribute all the files in the "personal category" she only needs to grant the key for the node "personal", which suddenly grants the deliver by the person the keys of all the big to small nodes ("photo", "music"). This is the same case, where most classes to be distributed belonging in to the same branch and the say super key of them is comfortable.

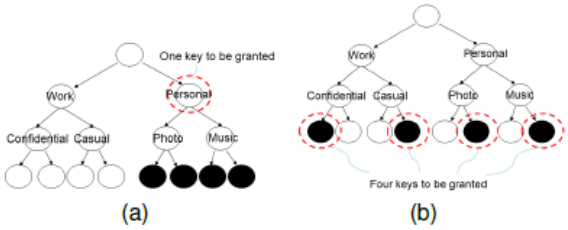


Fig 3 : Compact is not always possible for fixed hierarchy

However it is difficult for general cases as shown in fig:3(a), if Alice shares her demo music at work ("work"->"casual"->"demo" and "work"->"confidential"->"demo") with a partner who also has the rights to see some of her personal data, she can do to give more keys. Which raises an increasing the total key size one can see that this achieved is not comfortable when the classifications are more difficult and she wants to share different sets of files to different peoples for this the person to be deliver in our example the number of permitted secrete keys becomes the same as the number of classes. In general, hierarchical achieve can derive the problem individually if one estimate to share all files under a particular branch the hierarchy on aggregate, the number of keys increases the number of branches. It is not likely to come up with a hierarchy that can save the number of total keys to permitted for all individuals (which can admittance a different set of leaf nodes) alternate.

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

3.2 Compact key is Symmetric Key Encryption:

Encasing by the same problem of supporting comfortable hierarchy in decryption power deliver by the person (but in symmetric key setting), Benaloh et.al. [8] Presented an encryption program outside of the block schema which is originally proposed for concisely transmitting from one place to another place like large number of keys in broad cast scenario [18]. The arrangement is simple and we briefly to open its key equation process here for a concrete description of what are the comfortable property we want to approach. The equation of the key for a set of classes (which is a sub set of all possible cipher-text classes) is as follows a composite terms $N = p.q$ is chosen at alternate primes. A master secrete key Y is chosen at alternate from Z_N^* each class is linked with a distinct prime.

3.3 Compact Key in Identity-Based Encryption:

Identity-based encryption (IBE) (e.g., [20], [21], [22]) is a type of public-key encryption in which the public-key of a user can be used as an identity-string of the user (e.g., an email address). There is a belived party called private key generator (PKG) in IBE which hs a master-secret key and publishing a secret key to each user with respect to the user identity. The encryptor can take the public parameter and a user identity to encrypt a message. The receiver can decrypt this ciphertext by hi/her secret key. Guo et al. [23], [9] strived to build IBE with key aggregation. One of their schemes [23] assumes random oracles but another [9] does not. In their project, key aggregation is coerced in the sense that all keys to be aggregated must come from typical "identity divisions". While there are an exponential number of identities and thus secret keys, only a polynomial number of them can be aggregated. More importantly, their key-aggregation [23], [9] comes at the more cost of O(n) sizes for both Encoded text and the public parameter, where n is the number of secret keys which can be aggregated into a sustained size one. This increases more the costs of storing and transferring Encoded text, which is unsuitable in many situations such as shared cloud storage.

As we mentioned, our operations feature persistent ciphertext size, and their security holds in the persistent model. In fuzzy IBE [21], one single compact secret key can decrypt encoded texts encrypted under many identities which are close in a certain metric space, but not for an random set of identities and therefore it does not match with our idea of key aggregation.

3.4 Other Encryption Schemes:

Attribute-based encryption (ABE) [10], [24] allows each ciphertext to be joined with an attribute, and the master-secret key holder can remove a secret key for a guidelines of these characteristics so that a ciphertext can be decrypted by this key if its joined assign conforms to the guidelines. For example, with the secret key for the guidelines $(2 \vee 3 \vee 6 \vee 8)$, one can decrypt ciphertext attached with class 2, 3, 6 or 8. However, the major treat in ABE is connivance-hostility but not the compactness of secret keys. As expected, the size of the key frequently increases linearly with the number of attributes it encloses, or the ciphertext-size is not constant (e.g., [25]). To represent the decryption power of some ciphertexts without sending the secret key to the delegate, a useful unaffected is proxy re-encryption (PRE) (e.g., [26], [27],[28], [29]).

A PRE scheme allows Alice to unaffected to the server (proxy) the ability to convert the ciphertexts encrypted under her public-key into ones for Bob. PRE is well known to have numerous requisitions including cryptographic file system [30]. However, Alice has to believe the representative that it only converts ciphertexts according to her instruction, which is what we want to avoid at the first place. That also means that the transformation key of proxy should be well protected. Using PRE just moves the secure key storage requirement from the delegate to the attorney. It is thus undesirable to let the attorney occupy in the storage server. That will also be inefficient since every decryption requires separate interaction with the deputy.

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

4. Concrete Constructions of KAC:

4.1 The Basic Construction:

The basic scheme is designed from the "collusion-resistant broadcast encryption scheme " which was proposed by Boneh et al. This schema supports the constant size secret keys where each key only has the power to decrypt the cipher texts associated to particular index. For this, we utilize a new Extract algorithm & the corresponding Decrypt algorithm.

1. Setup(1^λ, n): Randomly pick a bilinear group G of prime order p where$2^\lambda \le p \le 2^{\lambda+1}$, a generator $g \in G$ and a $\in_R Z_p$. Compute $g_i = g^{\alpha^i} \in G$ for i=1,. . . ,n, n+2, . . . , 2n. Output the system parameter as param = $(g, g_1, \dots, g_n, g_{n+2}, \dots, g_{2n})$ (α will be deleted after Setup). Note that each ciphertext class is represented by an index in the integer set {1, 2, . . . , n}, where n is the maximum number of ciphertext classes.

2. KeyGen(): Pick $\gamma \in_R Z_p$, output the public and master-secret key pair: (pk = v = g^γ , msk = γ).

3.Encrypt(pk, i, m): For a message $m \in G_T$ and an index $i \in \{1,2,. . . .,n\}$, randomly pick $t \in_R Z_p$ and compute the ciphertext as $C = (g^t, (vg_i)^t, m.e(g_1, g_n)^t)$

4.Extract(msk =$^\gamma$, S): For the set S of indices j's, the aggregate key is computed as $K_s = \prod_{j\in S} g_{n+1-j}^{\gamma}$ Since S does not include 0, = $g_{n+1-j} = g^{\alpha^{n+1-j}}$ can always be retrieved from param.

5. Decrypt(K_s, S, i, C = {c1, c2, c3}): If i S, output $\perp$. Otherwise, return the message: $m = c_3.e(K_S.g_{n_1-j+i}, c_1)/e(\prod_{j\in S} g_{n+1-j}, c_2)$

For the data owner, with the knowledge of $^\gamma$, the term $e(g_1, g_n)^t$can be easily recovered by $e(c_1, g_n)^\gamma = e(g^t, g_n)^\gamma = e(g_1, g_n)^t$

For correctness, we can see that

$c_3.e(K_S.\prod_{j\in S, j\neq i} g_{n+1-j+i}, c_1)/e(\prod_{j\in S} g_{n+1-j}, g_i^t$

$= c_3.e(\prod_{j\in S, j\neq i} g_{n+1-j+i}, g^t)/e(\prod_{j\in S} g_{n+1-j}, g_i^t)$

$= c_3.\frac{e(\prod_{j\in S, g_{n+1-j+i}, g^t})\, e(g_{n+1}, g^t)}{e(\prod_{j\in S} g_{n+1-j+i}, g^t)}$

$= m.e(g_1, g_n)^t/e(g_{n+1}, g^t) = m$

4.1 Performance:

To do the encryption, the value $e^{(g1,gn)}$ is pre-computed and placed in the system parameter. From the other side, we observe that decryption takes only two pairings but only one of them is used in the aggregate key. Here we only use one pairing computation within the security chip storing the i.e, the aggregate key. This helps to compute the pairing fast in these present days, though the usage of resource constrained devices. The more flexible software implantations are also there for sensor nodes also.

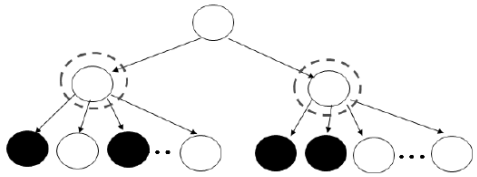


Fig: 4 Key Assignment in our Approch

4.2 Public-Key Extension:

To classify the cipher texts for more than one class(n), we need to register for additional key pairs i.e; (pk_2, msk_2),. . . , (pk_l, msk_l). Each class must be indexed by a 2-level index {(i,j)|1<=i<=l,1<=j<=n} where the number of classes ids increased by n when any key is added. Now the new public-key is treated as a new user, which have the concern i.e, the key aggregation is not possible for two independent users. If we face the problem of hierarchical solution as mentioned in Section 1, we can achieve still by using shorter key size and can gain flexibility as illustrated in Figure 4.It shows the flexibility for our approach. We can get the local aggregation, in which the secret keys from the same branch can be aggregated. For better explanation for our distinctive feature we use the 'quaternary tree' for the last level. In hierarchical approach when compared with quaternary trees the main advantage is preserved in which the delegates of the decryption power for all the 4 classes(if we choose

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

key for their parent class is delegated) or the number of keys will be same as the number of classes. Here in our approach at most 2 aggregate keys are needed to explain our example.

The details on how the encryption and decryption work when the public-key is extended is given below, which is somewhat similar to "$\sqrt{n}-$approach".

- Setup and KeyGen: Same as the basic construction.
- Extend(pk_l, msk_l): Execute KeyGen() to get$(v_{l+1}, \gamma_{l+1}) \epsilon G \times Z_p$, output the extended public and master-secret keys as $pk_{l+1} = (pk_l, v_{l+1}), msk_{l+1} = (msk_l, \gamma_{l+1})$

_ Encrypt$(pk_l, (a,b), m)$: Let $pk_l = \{v_1, \dots v_l\}$.For an index (a, b), $1 \le a \le l, 1 \le b \le n$,pickt $\in_R Z_p$, output the ciphertext as $C = (g^t, (v_a g_b)^t, m.e(g_1 g_n)^t)$

_ Extract$(msk_l; S_l)$: Let $msk_l = \{\gamma_1, \gamma_2, \dots, \gamma_l\}$. For a set$S_l$ of indices$(i,j), 1 \le i \le l, 1 \le j \le n$, get $g_{n+1-j} = g^{\alpha^{n+1-j}}$ from param, output:

$$K_{S_l} = \left(\prod_{(1,j)\in S_l} g_{n+1-j}^{\gamma 1}, \prod_{(2,j)\in S_l} g_{n+1-j}^{\gamma 2}, \dots, \prod_{(l,j)\in S_l} g_{n+1-j}^{\gamma l} \right).$$

_ Decrypt$(K_{S_l}, S_l, (a,b), C)$: If$(a,b) \notin S_l$, output $\perp$. Otherwise, let $K_{S_l} = (d_1, \dots, d_l)$ and $C = (c_1, c_2, c_3)$ Output the message: $m = \frac{c_3.e(d_a.\prod_{(a,j)\in S_l, j\neq b} g_{n+1-j+b}, c_1)}{e(\prod_{(a,j)\in S_l} g_{n+1-j}, c_2)}$

Just like the basic construction, the decryption can be done more efficiently with the knowledge of i's. Correctness is not much more difficult to see:

$$c_3.e(d_a.\prod_{(a,j)\in S_l, j\neq b} g_{n+1-j+b}, c_1)/e(\prod_{(a,j)\in S_l} g_{n+1-j}, c_2)$$

$$= c_3.e(\prod_{(a,j)\in S_l} g_{n+1-j}^{\gamma_a} . \prod_{(a,j)\in S_l, j\neq b} g_{n+1-j+b}, g^t) / e(\prod_{(a,j)\in S_l} g_{n+1-j+b}, (v_a g_b)^t)$$

$$= c_3.e(\prod_{(a,j)\in S_l, j\neq b} g_{n+1-j+b}, g^t)/e(\prod_{(a,j)\in S_l} g_{n+1-j}, g_b^t)$$

$$= m.e(g_1, g_n)^t / e(g_{n+1}, g^t) = m$$

We can also prove the semantic security of this extended scheme. The proof is very similar to that for the basic scheme and therefore is omitted. The public-key of our CCA construction to be presented below can also be extended using the same Extend algorithm.

4.2 Implication:

The extension approach can also be used for an update process as a key. If a secret value is occurred we can replace the occurred pk_1 with a new key pk_2.This small aggregate key size minimizes the communication over head in transferring the new key.

5 Performance Analysis:

5.1 Compression Factors:

When we have concrete comparison, we investigate about the requirements of space for the tree based key assignment approach as we discussed in Section 3.1. We use this in Complete Subtree scheme, which is a representative answer for the broadcast encryption problem which is also known as Subset-Cover framework. It involves a static-logical key hierarchy, which is madeup with a full binary key tree of height h(fig 3), and thus supports up to 2^h cipher text classes that is selected as a part of which is belonging to an authorized delegate. For any kind of ideal case as shown in figure 3(a), the Delegate is accessed to 2^h classes contain only one key, where hs is the height of a subtree .

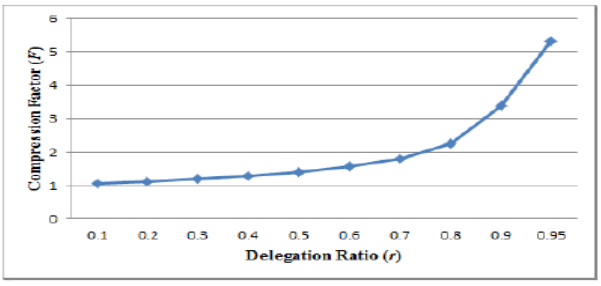
To decrypt cipher texts of a set of classes, it may have to hold large number of keys as depicted. So we concentrate in na where the number of symmetric keys to be assigned in this hierarchical key approach, in an average sense. Let us assume that there are exactly 2^h cipher text classes where the delegate of concern is entitled to 'r' of them i.e, r is the delegation ratio where it means the ratio of the delegated cipher text classes to the total classes.If r=0,n_a should also be 0,means no access to any of the classes ;if r=100%,na should be as low as 1, which means that the possession of only the root key in the hierarchy can grant the access to all the 2^h classes.

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

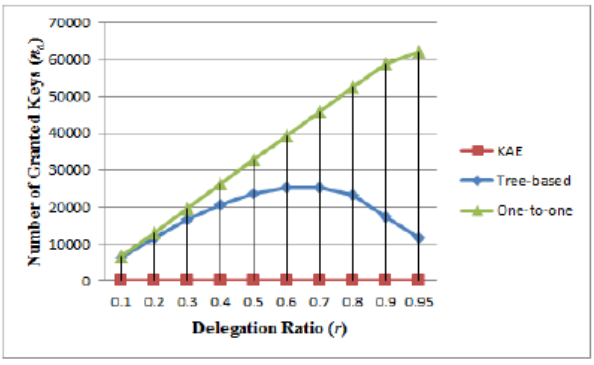
Consequently, one may expect that na may first increase with r, and may decrease later. We set r = 10%; 20%; _ _ _; 90%, and choose the portion in a random manner to model an arbitrary "delegation pattern" for different delegates. For each combination of r and h, we randomly generate 10^4 different combinations of classes to be delegated, and the output key set size na is the average over random delegations. We tabulate the results in Table 2, where h = 16; 18; 20 respectively. For a given h, na increases with the delegation ratio r until r reaches _ 70%. An amazing fact is that, the ratio of na to N($= 2^{h+1} - 1$), the total number of keys in the hierarchy (e.g., N = 15 in Figure 3), appears to be only determined by r but irrelevant of h. This is because when the number of ciphertext classes (2h) is large and the delegation ratio (r) is fixed, this kind of random delegation achieves roughly the same key.

h	r	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	0.95
16	n_a	6224.8	11772.5	16579.3	20545.8	23520.7	25263.8	25400.1	23252.6	17334.6	11670.2
	$\frac{n_a}{N}$	4.75%	8.98%	12.65%	15.68%	17.94%	19.27%	19.38%	17.74%	13.23%	8.90%
18	n_a	24895.8	47076.1	66312.4	82187.1	94078.8	101052.4	101594.8	93025.4	69337.4	46678.8
	$\frac{n_a}{N}$	4.75%	8.98%	12.65%	15.68%	17.94%	19.27%	19.38%	17.74%	13.23%	8.90%
20	n_a	99590.5	188322.0	265254.1	328749.5	376317.4	404205.0	406385.1	372085.2	277343.1	186725.4
	$\frac{n_a}{N}$	4.75%	8.98%	12.65%	15.68%	17.94%	19.27%	19.38%	17.74%	13.22%	8.90%

Table 2: Comparison ratio for different delegation ratios and heights



(a)



(b)

Fig 5: (a)Compression achieved by the tree-based approach for delegating different ratio of the classes (b) Number of granted keys (na) required for different approaches in the case of 65536 classes of data

5.2 Performance of Our Proposed Schemes:

Our attitude allow the consolidate factor F (F =n in our schemes) to be a adjustable framework, at the cost of O(n)-sized system framework. Encryption can be done in constant time, while decryption can be done in O(|S|) group multiplications (or point addition on elliptic curves) with 2 pairing operations, In which S is the set of cipher text(data that had been encrypted) classes decrypt table by the granted aggregate key and |S| ≤ n. As we assumed, key extraction requires O(|S|) group multiplications as well which seems to be not avoidable. However, as determined by the experiment results, we do not need to set a very high n to have better reduction than the tree-based proposal. Note that group multiplication is a very quick operation. Again, we confirm experimentally that our testing is true. We executed the basic KAC system in C with the Pairing-Based Cryptography (PBC) Library 8 version 0.4.18 for the underlying elliptic-curve group and pairing functioning.

Since the permitted key can be as small as one G element, and the ciphertext only contains G and one GT elements, we used (symmetric) pairings over Type-A (super singular) curves as proposed in the PBC library which contains the highest coherence among all types of curves, even though Type-A curves do not provide the shortest depiction for group elements. In our application, p is a 160-bit stepladder prime, which offers 1024-bit of discrete-logarithm security. With this Type-A curves setting in PBC, elements of groups G and GT take 512 and 1024 bits to represent, respectively. The test machine is a Sun UltraSparc IIIi system with dual CPU (1002 MHz) running Solaris, each with 2GB RAM. The timings reported below are averaged over 100 irregular runs. In our test, we take the number of ciphertext classes n =

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

2^{16} = 65536. The Setup algorithm, while outputting (2n + 1) elements by doing (2n − 2) exponentiations, can be made logical by preprocessing functionality by PBC, which reduces time for exponentiation the same element (g) in the long run. This is the only "low-level" minimization trick we have used. All other operations are executed in direct order. In particular, we did not take advantage of the fact that ˆ e(g1, gn) will be exponented many times across different encryptions. However, we pre-calculated its value in the setup stage, such that the encryption can be done without computing any pairing. Our test results are shown in Table 3. The execution times of Setup, KeyGen, Encrypt are independent of the entrusting ratio r. In our tests, KeyGen takes 3.3 milliseconds and Encrypt takes 6.8 milliseconds. As we assumed, the running time complexities of Extract and Decrypt increase linearly with the delegation ratio r(which determines the size of the delegated set S). Our timing results also depend on what can be seen from the expression in Extract and Decrypt — two pairing operations take negligible time, the running time of Decrypt is approximately a double of Extract.

Note that our experiments deal with up to 65536 number of classes (which is also the compression factor), and should be large sufficient for fine-grained data transferring in most situations. Finally, we remark that for applications where the number of ciphertext classes is large but the non confidential storage is limited, one should employ our schemes using the Type-D pairing bundled with the PBC, which only requires 170-bit to represent an element in G. For n = 216, the system limit requires approximately 2.6 megabytes, which is as large as a lower feature MP3 file or a higher-determination JPEG file that a typical cellphone can store more than a dozen of them.But we saved cost secure storage without the argument of managing a hierarchy of entrusting classes.

6. New Patient-Controlled Encryption:

Stimulated by the nationwide effort to computerize America's medical records, the concept of patient controlled encryption (PCE) has been studied. In PCE, the health record is separate into a different forms based on the use of different ontology (metaphysical science), and patients are the clients who will produce and use secret data. When there is a need for a healthcare personnel to know the part of the record, a patient will say the secret information for the particular part of the record. In the work of Benaloh , three solutions have been provided, which are symmetric-key PCE for fixed hierarchy (the "folklore" tree-based method in Section 3.1), public-key PCE for fixed hierarchy (the IBE analog of the folklore method, as mentioned in Section 3.1), and RSA-based symmetric-key PCE for "flexible hierarchy" (which is the "set membership" access policy as we explained). Our work provides a candidate relevant information for the missing piece, public-key PCE for flexible hierarchy, which the containers of an efficient data was an open query. Any patient can either revel her own hierarchy according to her need, or follow the set of types suggested by the electronic medical record system she/he is using, such as "clinic visits", "x-rays", "allergies", "medications" and so on. When the patient wishes to give rights to her doctor to acess, she/he can select any subset of these categories and issue a single key, from which keys for all these categories can be calculated.

Fig : Login Page

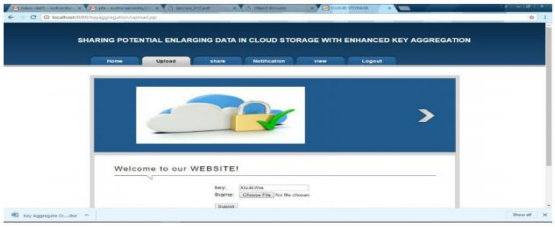


Fig : Upload Page

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

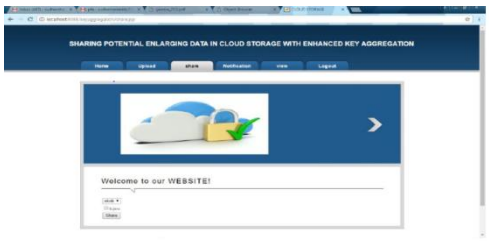


Fig: Share Page

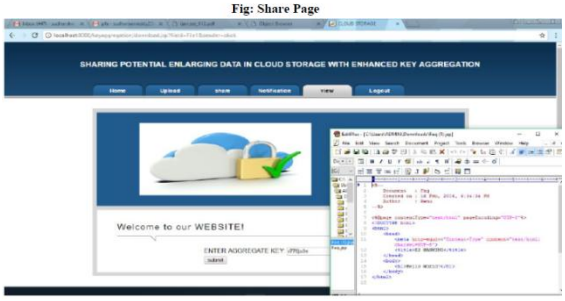


Fig: Viewing Page

7. Conclusion and Future Work:

How to protect users' information is a central question of cloud storage. With more mathematical tools, cryptographic schemes are having more multiples and also involve multiple keys for a single application. In this article, we consider how to "compress" secret keys in public-key cryptosystems which having delegation of secret keys for different ciphertext classes in cloud storage. No matter which one among the power set of classes, the delegate can always get an aggregate key of constant size. Our view is more flexible than hierarchical key assignment which can only save spaces if all key-holders share a similar set of dispensation. A deficiency in our work is the predefined way of the number of maximum ciphertext classes. In cloud storage, the number of ciphertexts usually grows more rapidly. So we have to reserve enough ciphertext classes for the future use. Otherwise, we need to expand the public-key as we described in Section 4.2.Although the parameter can be downloaded with ciphertexts, it would be better if its size is independent of the maximum number of ciphertext classes. On the other hand, when one carries the delegated keys around in a mobile device without using special trusted hardware, the key is dispose to revel, designing a leakage resilient cryptosystem [22], [34] yet allows efficient and flexible key delegation is also an interesting .

8. References:

1. "Key-Aggregate Cryptosystem for Scalable Data Sharing in Cloud Storage", Cheng-Kang Chu, Sherman S. M. Chow, Wen-Guey Tzeng, Jianying Zhou, and Robert H. Deng, Senior Member, IEEE

2. L. Hardesty, "Secure computers aren't so secure," MIT press, 2009, http://www.physorg.com/news176107396.html.

3. C. Wang, S. S. M. Chow, Q. Wang, K. Ren, and W. Lou, "Privacy- Preserving Public Auditing for Secure Cloud Storage," IEEE Trans. Computers, vol. 62, no. 2, pp. 362–375, 2013.

4. B. Wang, S. S. M. Chow, M. Li, and H. Li, "Storing Shared Data on the Cloud via Security-Mediator," in International Conference on Distributed Computing Systems - ICDCS 2013. IEEE, 2013.

5. 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 - Essays Dedicated to Jean-Jacques Quisquater on the Occasion of His 65th Birthday, ser. LNCS, vol. 6805. Springer, 2012, pp. 442–464.

6. D. Boneh, C. Gentry, B. Lynn, and H. Shacham, "Aggregate and Verifiably Encrypted Signatures from Bilinear Maps," in Proceedings of Advances in Cryptology - EUROCRYPT '03, ser. LNCS, vol. 2656. Springer, 2003, pp. 416–432.

7. M. J. Atallah, M. Blanton, N. Fazio, and K. B. Frikken, "Dynamic and Efficient Key Management for Access Hierarchies," ACM Transactions on Information and System Security (TISSEC), vol. 12, no. 3, 2009.

8. J. Benaloh, M. Chase, E. Horvitz, and K. Lauter, "Patient Controlled Encryption: Ensuring Privacy of Electronic Medical Records," in Proceedings of ACM

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

Workshop on Cloud Computing Security (CCSW '09). ACM, 2009, pp. 103–114.

9. F. Guo, Y. Mu, Z. Chen, and L. Xu, "Multi-Identity Single-Key Decryption without Random Oracles," in Proceedings of Information Security and Cryptology (Inscrypt '07), ser. LNCS, vol. 4990. Springer, 2007, pp. 384–398.

10. S. S. M. Chow, Y. J. He, L. C. K. Hui, and S.-M. Yiu, "SPICE -Simple Privacy-Preserving Identity-Management for Cloud Environment," in Applied Cryptography and Network Security – ACNS 2012, ser. LNCS, vol. 7341. Springer, 2012, pp. 526–543.