© 2017 IJSRST | Volume 3 | Issue 1 | Print ISSN: 2395-6011 | Online ISSN: 2395-602X
Themed Section: Science and Technology

Improved Shamir's Secret Based Key Aggregate Mechanism for Secure Data Sharing in Multi Cloud Storage

Swapnali Vilas Arote, Prof R. L. Paikrao

Computer Engineering Department, Amrutvahini College of Engineering, Sangamner, Maharashtra, India

ABSTRACT

The Data sharing is an important functionality in cloud storage. We describe new public key crypto systems which produce constant-size cipher texts such that efficient delegation of decryption rights for any set of cipher texts are possible. The novelty is that one can aggregate any set of secret keys and make them as compact as a single key, but encompassing the power of all the keys being aggregated. Ensuring the security of cloud computing is second major factor and dealing with because of service availability failure the single cloud providers demonstrated less famous failure and possibility malicious insiders in the single cloud. A movement towards Multi-Clouds, In other words "Inter-Clouds" or "Cloud-Of-Clouds" as emerged recently. This works aim to reduce security risk and better flexibility and efficiency to the user. Multi-cloud environment has ability to reduce the security risks as well as it can ensure the security and reliability.

Keywords: Cloud Storage, Key Aggregate Encryption, Multi-cloud infrastructure, Data Sharing.

I. INTRODUCTION

Cloud storage is gaining popularity recently. In enterprise settings, we see the rise in demand for data outsourcing, which assists in the strategic management of corporate data. It is also used as a core technology behind many online services for personal applications. Nowadays, it is easy to apply for free accounts for email, photo album, file sharing and/or remote access, with storage size more than 25GB (or a few dollars for more than 1TB). Together with the current wireless technology, users can access almost all of their files and emails by a mobile phone 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, which means any unexpected privilege escalation will expose all data. In a shared-tenancy cloud computing environment, things become even worse. Data from different clients can be hosted on separate virtual machines (VMs) but reside on a single physical machine. Data in a target VM could be stolen by instantiating another VM co-resident with the target one. Regarding availability of files, there are a series of cryptographic schemes which go as far as allowing a third-party auditor to check the availability of files on behalf of the data owner without leaking anything about the data, or without compromising the data owners anonymity. Likewise, cloud users probably will not hold the strong belief that the cloud server is doing a good job in terms of confidentiality. A cryptographic solution, with proven security re-lied on number-theoretic assumptions is more desirable, whenever the user is not perfectly happy with trusting the security of the VM or the honesty of the technical staff. These users are motivated to encrypt their data with their own keys before uploading them to the server. Data sharing is an important functionality in cloud storage. For example, bloggers can let their friends view a subset of their private pictures; an enterprise may grant her employees access to a portion of sensitive data. The challenging problem is how to effectively share encrypted data. of course users can download the encrypted data from the storage, decrypt them, then send them to others for sharing, but it loses the value of cloud storage. Users should be able to delegate the access rights of the sharing data to others so that they can access these data from the server.

II. METHODS AND MATERIAL

1. Related Work

A. Identity Bases Encryption (IBE)

IBE is a type of a public-key encryption. Identity string is set for encryption which is nothing but users public key. In IBE, master secret keys are generated by the private key generator and here the secret key is provided based on users identity. Sender wants to share files. So sender will encrypt the files by making use of user identity and public parameter and sends the files. Receiver will decrypt these files by making use of his secret key. But out of key-aggregation and IBE, only one assumes random oracles. Key aggregation is inhibited as keys to be aggregated will come from various identity. The disadvantages of this system is ciphertext size is non-constant and cost of storing ciphertext and transmitting it expensive.

B. Symmetric Key Encryption

Benaloh proposed an encryption scheme, where a huge number of keys can be sent rapidly in a broadcast scenario. The key origin is as follows. Initially choose two prime numbers p and q for a composite module. At random, master secret key will be chosen. Dissimilar prime numbers will be allied with each class. A public system parameter is considered for which all the prime numbers will be put. The outcome of this is a constant size key. This method is designed for symmetric-key setting. So here the sender should encrypt files with corresponding secret keys which will not be feasible. The disadvantages of this system are both encryption and decryption is done by same key and encryptor should get corresponding key to encrypt files.

C. Attribute Based Encryption (ABE)

In Attribute Based Encryption method an attribute will be linked with cipher text. From master secret key, the secret key will be derived. This secret key is used to decrypt the files merely if all its connected attributes go after the rules. Before Attribute Based Encryption method was introduced, the user who wanted secret key must go to third party and proving he is real by providing his identity and then he was capable to decrypt the file. Later in ABE scheme the secret key of user was not allowed to a single centre. Instead it was authorized by independent authorities. But still this scheme has drawback i.e. no solidity on secret key. Here in this scheme there is linear increase in key size, with the increase in attributes. Disadvantages are (a) Decryption key size is non-constant. (b) Requires more space to store keys. (c) Decryption key size increases linearly. (d) Managing keys is expensive.

D. Attribute-Based Encryption for Fine-Grained Access Control of Encrypted Data

Attribute-based encryption (ABE)allows each ciphertext to be associated with an attribute, and the master-secret key holder can extract a secret key for a policy of these attributes so that a ciphertext can be decrypted by this key if its associated attribute conforms to the policy. For example, with the secret key for the policy (2v3v6v8), one can decrypt ciphertext tagged with class 2, 3, 6, or 8. However, the major concern in ABE is collusion resistance but not the compactness of secret keys. Indeed, the size of the key often increases linearly with the number of attributes it encompasses, or the ciphertext-size is not constant.

E. Chosen-Ciphertext Secure Proxy Re-Encryption

To delegate the decryption power of some ciphertexts without sending the secret key to the delegatee, a useful primitive is proxy re-encryption (PRE). A PRE scheme allows sender to delegate to the server(proxy) the ability to convert the ciphertexts encrypted under her public-key into ones for receiver. PRE is well known to have numerous applications including cryptographic file system. Nevertheless, sender has to trust the proxy that it only converts ciphertexts according to her instruction ,which is what we want to avoid at the first place. Even worse, if the proxy colludes with receiver, some form of senders secret key can be recovered which can decrypt senders(convertible) ciphertexts without receivers further help. That also means that the transformation key of proxy should be well protected. Using PRE just moves the secure key storage requirement from the delegatee to the proxy. It is, thus, undesirable to let the proxy reside in the storage server. That will also be inconvenient since every decryption requires separate interaction with the proxy. Using PRE just moves the secure key storage requirement from the delegatee to the proxy. It is thus, undesirable to let the proxy reside in the storage server. That will also be inconvenient since every decryption requires separate interaction with the proxy.

F. Dynamic and Efficient Key Management for Access Hierarchies

We start by discussing the most relevant study in the literature of cryptography/security. Cryptographic key assignment schemes aim to minimize the expensive storing and managing secret keys for general cryptographic use. Utilizing a tree structure, a key Using KAC for data sharing in cloud storage. We call this as master-secret key to avoid confusion with the delegated key we will explain later. For simplicity, we omit the inclusion of a decryption algorithm for the original data owner using the mastersecret key. In our specific constructions, we will show how the know ledge of the master-secret key allows a faster decryption than using Extract followed by Decrypt. For a given branch can be used to derive the keys of its. Just, granting the parent key implicitly grants all the keys of its descendant nodes. proposed a method to generate a tree hierarchy of symmetric-keys by using repeated evaluations of pseudo random function/block-cipher on a fixed secret. The concept can be generalized from a tree to a graph. More advanced cryptographic key assignment schemes support access policy that can be modelled by an acyclic graph or a cyclic graph . Most of these schemes produce keys for symmetric-key cryptosystems, even though the key derivations may require modular arithmetic as used in public-key cryptosystems, which are generally more expensive than symmetric-key operations such as pseudorandom function. We take the tree structure as an example. Alice can first classify the ciphertext classes according to their subjects like Each node in the tree represents a secret key, while the leaf nodes represents the keys for individual ciphertext classes. Filled circles repre-sent the keys for the classes to be delegated and circles circumvented by dotted lines represent the keys to be granted. Note that every key of the non leaf node can derive the keys of its descendant nodes. if Alice wants to share all the files in the personal category, she only needs to grant the key for the node personal, which automatically grants the delegatee the keys of all the descendant nodes (photo, music). This is the ideal case, where most classes to be shared belong to the same branch and thus a parent key of them is sufficient.

G. Fuzzy Identity-Based Encryption. Theory and Applications of Cryptographic Techniques:

IBE is a type of public-key encryption in which the public-key of a user can be set as an identity-string of the user (e.g., an email address). There is a trusted party called private key generator in IBE which holds a master-secret key and is sues 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 recipient can decrypt this ciphertext by his secret key. Guest tried to build IBE with key aggregation. One of their schemes assumes random oracles but another does not. In their schemes, key aggregation is constrained in the sense that all keys to be aggregated must come from different identity divisions. While there are an exponential number of identities and thus secret keys, only a polynomial number of them can b e aggregated. Most importantly, their key aggregation comes at the expense of On sizes for both ciphertext and the public parameter, where is the number of secret keys which can be aggregated into a constant size one. This greatly increases the costs of storing and transmitting ciphertext, which is impractical in many situations such as shared cloud storage. As we mentioned, our schemes feature constant ciphertext size, and their security holds in the standard model. In fuzzy IBE, one single compact secret key can decrypt ciphertext encrypted under many identities which are close in a certain metric space.

There exist several expressive ABE schemes where the decryption algorithm only requires a constant number of pairing computations. Recently, Green et al. Proposed a remedy to this problem by introducing the notion of ABE with outsourced decryption, which largely eliminates the decryption overhead for users. Based on the existing ABE schemes Green et al. Also presented concrete ABE scheme with outsourced decryption. ABE cipher text CT satisfied by that users attributes or access policies into a simple cipher text CT and it only incurs a small overhead for the user to recover the plain text from transformed cipher text CT. The security property of the ABE scheme with outsourced decryption guarantees that an adversary (including the malicious server) be not able to learn anything about the encrypted message;

however, the scheme provides no guarantee on the correctness of the transformation done by the cloud server. In the cloud computing setting, cloud service providers may have strong financial incentives to return incorrect answers, if such answers requires less work and are unlikely detected by users. Drawbacks of Existing System are Symmetric Key Encryption, Key-Policy Attribute based Encryption, Relation between class required, Single cloud is merged, No guarantee on staff.

2. Proposed System

The data owner establishes the public system parameter via Setup and generates a public/master-secret key pair via KeyGen. Messages can be encrypted via Encrypt by anyone who also decides what ciphertext class is associated with the plaintext message to be encrypted. The data owner can use the master-secret to generate an aggregate decryption key for a set of ciphertext classes via Extract. The generated keys can be passed to delegates securely (via secure e-mails or secure devices) Finally, any user with an aggregate key can decrypt any ciphertext provided that the ciphertexts class is contained in the aggregate key via Decrypt.

The key aggregation property is especially useful when we expect the delegation to be efficient and flexible. The schemes enable a content provider to share her data in a Confidential and selective way, with a fixed and small ciphertext expansion, by distributing to each authorized user a single and small aggregate key. Here, we describe the main idea of data sharing in cloud storage using KAC, illustrated in Fig. Suppose User1 wants to share her data m1;m2;.....;m on the server. She first performs Setup1; n to get param and execute KeyGen to get the public/mastersecret key pair pk; msk. The system parameter param and public-key pk can be made public and master-secret key msk should be kept secret by User1. Anyone (including User1 herself) can then encrypt each mi by Ci Encryptpk; i; mi. The encrypted data are uploaded to the server. With param and pk, people who cooperate with User1 can update User1s data on the server. Once User1 is willing to share a set S of her data with a friend User2, He can compute the aggregate key KS for User2 by performing Extractmsk; S. Since KS is just a constant-size key, it is easy to be sent to User2 via a secure e-mail. After obtaining the aggregate key, User2 can download the data he is authorized to access. That is, for each i 2 S, User2 downloads Ci (and some needed values in param) from the server. With the aggregate key KS, User2 can decrypt each Ci by DecryptKS; S; i; Ci for each i 2 S.

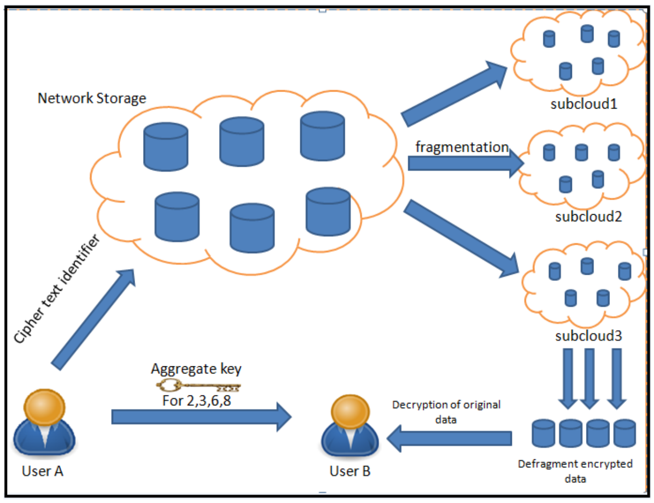

Figure 1: System Architecture

A. Aggregate Key Generation Algorithm

Input : Secret data
Output : Aggregate Key
Step 1: Upload Secret
Step 2: Generate Key (SHA-1)
Step 3: AES Encryption
Step 4 :Authority Delegation
Step 5: Secret key i.e. S
Step 6: $f(x)=(n1x + n2\ x^2+n3\ x^3+ S)$
Step 7: Generate Aggregate Key
Step 8:File Fragmentation
Step 8:File Reconstruction
Step 9: Encrypted Search Mechanism
Step 10: Decrypt File and Search

B. Shamir Secrete Sharing Algorithm:

Step 1. Suppose that our secret is 1234(s=1234).
Step 2. We wish to divide the secret into 6 parts (n=6),where any subset of 3 parts (k=3) is sufficient to reconstruct secret.
Step 3. We select 2 random numbers: 166, 94. (a1=166, a2=94). Our polynomial to produce secret shares (points) is therefore:

$$f(x)=1234 + 166x+ 94x^2$$

Step 4. We construct 6 points from the polynomial: (1,1494),(2,1942),(3,2578),(4,3402),(5,4414),(6, 5614). We give each participant a different single point.

C. Example

Suppose that our secret is 1234 i.e. S =1234.
We wish to divide the secret into 6 parts i.e. n=6, where any subset of 3 parts .i.e k=3 is sufficient to reconstruct the secret. At random we obtain 2 (k-1) numbers: 166 and 94.

$$(a_1 = 166,\ a_2 = 94)$$

Our polynomial to produce secret shares (points) is therefore:

$$f(x) = 1234 + 166x + 94x^2$$

We construct 6 points from the polynomial:
(1,1494),(2,1942),(3,2578), (4,3402),(5,4414),(6,5614)
We give each participant a different single point (both x and f(x)).

Reconstruction

In order to reconstruct the secret any 3 points are enough.

Let us consider

(x0,y0) =(2,1942);(x1,y1) = (4,3402); (x2,y2) = (5,4414)

We compute Lagrange basis polynomials:

$$\ell_0 = \frac{x-x_1}{x_0-x_1}\cdot\frac{x-x_2}{x_0-x_2} = \frac{x-4}{2-4}\cdot\frac{x-5}{2-5} = \frac{1}{6}x^2 - \frac{3}{2}x + \frac{10}{3}$$
$$\ell_1 = \frac{x-x_0}{x_1-x_0}\cdot\frac{x-x_2}{x_1-x_2} = \frac{x-2}{4-2}\cdot\frac{x-5}{4-5} = -\frac{1}{2}x^2 + \frac{7}{2}x - 5$$
$$\ell_2 = \frac{x-x_0}{x_2-x_0}\cdot\frac{x-x_1}{x_2-x_1} = \frac{x-2}{5-2}\cdot\frac{x-4}{5-4} = \frac{1}{3}x^2 - 2x + \frac{8}{3}$$

Therefore:

$$f(x) = \sum_{j=0}^{2} y_j \cdot \ell_j(x)$$
$$= 1942\cdot\left(\frac{1}{6}x^2 - \frac{3}{2}x + \frac{10}{3}\right) + 3402\cdot\left(-\frac{1}{2}x^2 + \frac{7}{2}x - 5\right) + 4414\cdot\left(\frac{1}{3}x^2 - 2x + \frac{8}{3}\right)$$
$$= 1234 + 166x + 94x^2$$

Shamir's secret sharing scheme

Shamir's secret sharing scheme is based on polynomial evaluations. The central party is the dealer that performs share computation operations on input secrets and distributes the resulting shares to other parties. When the secret has to be reconstructed, the parties give their shares to the dealer, which can then combine the shares and retrieve the secret.

In Shamir's scheme shares are evaluations of a randomly generated polynomial. The polynomial f is generated in such a way that the evaluation f (0) reveals the secret value. If there are enough evaluations, the parties can reconstruct the polynomial and compute the secret. Algorithm 1 describes how shares are commutated in Shamir's scheme.

```
Algorithm 1: Share computation algorithm for Shamir's scheme
  Data: finite field $\mathbf{F}$, secret data $s \in \mathbf{F}$, threshold $k$, number of shares $n$
  Result: shares $s_1, \ldots, s_n$
  Set $f_0 = s$
  Uniformly generate coefficients $f_1, \ldots, f_{k-1} \in \mathbf{F}$
  Construct the polynomial $f(x) = f_0 + f_1 x + \cdots + f_{k-1} x^{k-1}$
  Evaluate the polynomial: $s_i = f(i)$, $(i = 1, \ldots, n)$
```

Note that the indices of the shares start from one, as the author cannot output s0 = f (0), because it is the secret value. The resulting shares $s_1 \ldots s_n$ can be distributed to their holders. If the original value needs to be retrieved, they need a subset of at least k shares. Note that it is important to store the index i together with the share s_i, because it is later needed for reconstruction.

The classical algorithm of Shamir's scheme reconstructs the whole polynomial, whereas they describe versions optimized for reconstructing only the secret f (0) = s. They only need to compute f (0) so for our purposes we can simplify the base polynomial's bi(x) as follows:

$$\beta_i = b_i(0) = \prod_{\substack{j=1 \\ i \neq j}}^{k} \frac{(-a_j)}{(a_i - a_j)}.$$

If the shares are computed using Shamir's scheme then algorithm 2 retrieves the secret value s.

```
Algorithm 2: Share reconstruction algorithm for Shamir's scheme
  Data: finite field F, shares $s_{t_1}, \ldots, s_{t_k} \in F$ where $t_j \in \{1, \ldots, n\}$ are
        distinct indices
  Result: secret data $s$
  compute the reconstruction coefficients $\beta_i$ according to equation (3)
  compute $f(0) = s_{t_1}\beta_{t_1} + \cdots + s_{t_k}\beta_{t_k}$
  Return $s = f(0)$
```

Secure computation with shares

They will now show what can be done with the shares once they have been distributed. They will investigate the possibility of using the homomorphic property of the secret sharing scheme to perform operations with the shares. In the following assume that a k-out-of-n threshold scheme is used. Assume that we have n parties $p_{1....}p_n$ and the dealer gives each one of them a share according to its index.

Addition

Assume that we have shared values [u] = [$u_{1......}$ u_n] and [v] = [$v_{1......}v_n$]. Because the evaluation mapping is a linear transformation, they can add the shares of [u] and [v] to create a shared value [w] so that u + v = w. Each party k has to run the protocol given in Algorithm 3 to add two shared values.

```
Algorithm 3: Protocol for adding two Shamir shares for node $k$
  Data: shares $u_k$ and $v_k$
  Result: share $w_k$ that represents the sum of $[u]$ and $[v]$
  Round 1
    $w_k = u_k + v_k$
```

Multiplication with a scalar

Assume that we have a shared value [u] = [u_1….. u_n] and a public value t. They can multiply the shares u_i with t so that the resulting shares represent the value [w] = t[u]. Algorithm 4 shows the protocol for multiplication a share value by a scalar.

```
Algorithm 4: Protocol for multiplying Shamir shares by a scalar value for
node $k$
  Data: shares $u_k$ and a public value $t$
  Result: share $w_k$ that represents the value of $t[u]$
  Round 1
    $w_k = tu_k$
```

Multiplication

Assume that they have shared values [u] = [$u_{1........}$ u_n] and [v] = [v1……v_n]. Share multiplication, unfortunately, cannot be solved with the linear property of the transformation, as multiplying two polynomials with the same degree gives a polynomial with double the degree of the source polynomials. This means that they must use a k-out-of-n threshold scheme where 2k ≤ n and the polynomials must have a degree of at most 2k. By multiplying the respective shares, the miners actually compute a share that represents the polynomial storing the product of the secrets. However, they must reconstruct the secret stored in the product polynomial and reshare it to make further multiplications possible.

Otherwise, the multiplication of the product polynomial with another one will give us a polynomial with a degree larger than n and we cannot reconstruct the secret from such polynomials anymore.

They can use precomputed values of the optimized base polynomials βi needed in the protocol. This requires each node to know its number and also how many other nodes there are, but that is a reasonable assumption. Algorithm 5 gives the complete protocol for multiplying Shamir shares.

```
Algorithm 5: Protocol for multiplying two Shamir shares for node $i$
  Data: shares $u_i$ and $v_i$, precomputed value $\beta_i$
  Result: share $w_i$ that represents the value of $[u][v]$
  Round 1
    $z_i = u_i v_i \beta_i$
    Share $z_i$ to $z_{i_1}, \ldots, z_{i_n}$ using the same scheme as the dealer uses
    Send to each other node $P_l, i \neq l$ the share $z_{i_l}$
  Round 2
    Receive shares $z_{ji}, j \neq i$ from other nodes
    $w_i = z_{i_i} + \sum_{j=1, j \neq i}^{n} z_{ji}$
```

III. RESULTS AND DISCUSSION

A. Improved KAC system vs KAC system

Aggregate Key Generation Time comparison with varying number of files.

Table I : Time Comparison

No of Files	KAC System(Time in ms)	Improved KAC System(Time in ms)
1	20.979	21.918
5	68.573	52.742
10	16.4350	100.939
20	187.350	115.300
100	256.347	130.634

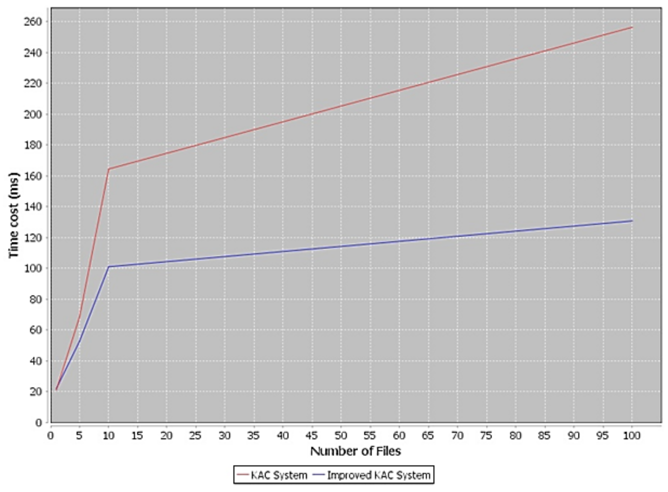

Figure 2. Time Generation of AGK generation

B. SHA-1 (Hash Function) Key Size Comparison

In improved KAC System we use SHA-1 128 bit hash function to encrypt the key of uploaded file instead of SHA-1 256 bit encryption. Due to this the system Time performance is much improved as the Improved KAC system time is reduced for encryption as well as decryption process. As we are using Shamir secret in next stage for file fragmentation, therefore the SHA-1 hash function prediction for intruder is not possible as file is uploaded in parts on cloud.

Table II. Key Size Comparison

No of Files	KAC System(AG key Size)in bits	Improved KAC System (AG key Size) in bits
1	256	128
5	1280	640
10	2560	1280
20	5120	2560
100	25600	12800

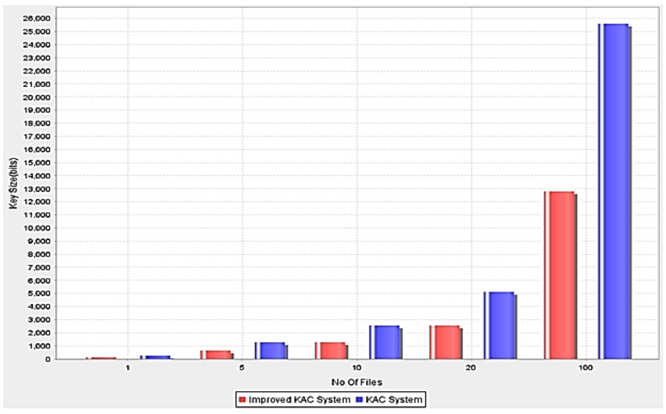

Figure 3. SHA-1 Key Comparison

C. Shamir Secrete Algorithm Improvement:

Shamir Secret Algorithm Runs On Numeric Values So we converted the encrypted data of file bundle assigned to aggregate key into the numeric format using java sandmark utility.

In Java we have Big decimal numeric object data type which can hold the 2^35 digits and the big decimal size is 32-bits. hence if we consider a string of 256 characters for 1 character size in java in 2 bytes i.e. 512 bits memory for string but using the above mechanism we reduce the file size to 32 bits.

now in improved shamir secrete algorithm we use a linear equation instead of quadratic equation hence the time performance of system is improved as the we require only 2 **Lagrange polynomials** instead of 3 which reduces precious server processing time for reconstruction of files.

Table III. Time Performance

No of Files	KAC System(ms)	Improved KAC System(ms)
1	0	0
5	6.200	3.700
10	11.978	11.930
20	27.056	24.056

D. Improved Shamir Secrete Algorithm:

1234, and can only be revealed when two people combine their information. For this we thus need a linear equation, such as:

$$Y = 2x + 1234$$

1234, and can only be revealed when two people combine their information. For this we thus need a linear equation, such as:

(1,1236)and(2,1238)

If we only know one point, such as (1,1236), we cannot determine the equation of the line, but knowing two points we can determine the gradient:

Gradient = $(y2-y1)/(x2-x1) = (1236 -1238)/(2-1) = 2$
and from here we can determine the point it cuts the y-axis (c):

$$y = mx+c$$
$$c = y - mx = 1236 - 2^{*} 1 = 1234$$

and thus we have the equation ($y = 2x + 1234$), where the secret is 1234.

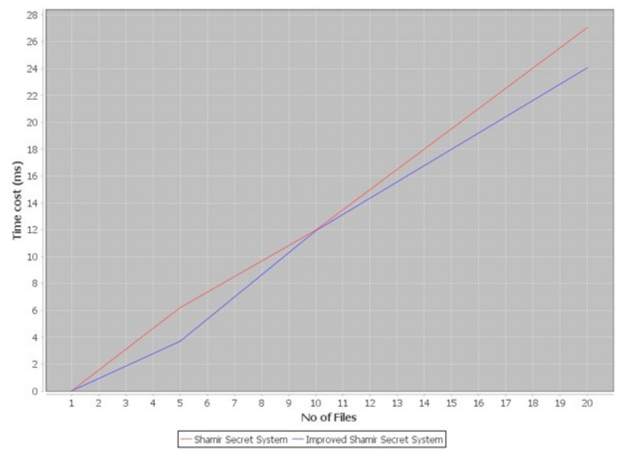

Figure 4. Time cost of fragmentation

IV. CONCLUSION

In this work we have reviewed three authentication techniques: Attribute based encryption (ABE), Identity Based Encryption (IBE) and Key Aggregate Cryptosystem (KAC). The major concern in ABE is collusion resistance but not compression of secret keys. Definitely, the ciphertext size is not constant. In IBE, random set of identities are not match with our design of key aggregation. Key Aggregate Cryptosystem protects users data privacy by compressing the secret key in public key cryptosystem which supports delegation of secret key for different cipher text classes. For future extension it is necessary to reserve enough cipher texts classes because in cloud cipher texts grows rapidly and the limitation is that bound of the number of maximum cipher text classes. To share data exibly is vital thing in cloud computing. Users prefer to upload there data on cloud and among different users. Outsourcing of data to server may lead to leak the private data of user to everyone. Encryption is a one solution which provides to share selected data with desired candidate. Sharing of decryption keys in secure way plays important role. Public-key cryptosystems provides delegation of secret keys for different ciphertext classes in cloud storage. There is some limitation to the existing system like predefined bound of the number of maximum ciphertext classes and system is prompt to leakage of key.

V. REFERENCES

[1]. F. C. Chang and H. C. Huang, \Key-Aggregate Cryptosystem for Scalable Data Sharing in Cloud Storage," Inf. Sci., vol. 192, no. 1, pp. 3949, Jun. 2012.

[2]. 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. 526543. pp. 173184, 2011.

[3]. L. Hardesty, Secure computers arent so secure, \MIT press,"2009,http://www.physorg.com/news176107396.html..

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

[5]. S.S.M. Chow, Y. Dodis, Y. Rouselakis, and B. Waters, \Practical Leakage- Resilient Identity-Based Encryption from Simple Assumptions ," in Proc. ACM Conf. Com-puter and Comm. Security,pp. 152-161, 2010.

[6]. G. Ateniese, A.D. Santis, A.L. Ferrara, and B. Masucci, \Provably-Secure Time-Bound Hierarchical Key Assignment Schemes, ,"J. Cryptology.,vol. 25, no. 2, pp. 243-270, 2012.

[7]. F. Guo, Y. Mu, Z. Chen, and L. Xug, \Multi-Identity Single-Key Decryption without Random Oraclesl,",in Proceedings of Information Security and Cryptology (Inscrypt 07), ser. LNCS, vol. 4990. Springer, 2007, pp. 384398.

[8]. Vigneshwaran.K 1, Sumithra.S2, Janani.R3 "An Intelligent Tracking System Based on GSM and GPS Using Smartphones" Vol. 4, Issue 5, May 2015.