Available Online at www.ijcsmc.com

International Journal of Computer Science and Mobile Computing

A Monthly Journal of Computer Science and Information Technology

ISSN 2320–088X

IJCSMC, Vol. 4, Issue. 4, April 2015, pg.906 – 923

RESEARCH ARTICLE

Data Security in Cloud Using Aggregate Key and Diffie-Hellman Algorithm

Mrs. Savita Kacheshwar Dabhade, ME Computer Engineering
Prof. M.K.Kshirsagar, Computer Engineering
Vishwabharti Academy COE, Ahmednagar, Pune University

Abstract:

Cloud technology is very constructive and useful in present new technological era, where a person uses the internet and the remote servers to give and maintain data as well as applications. Such applications in turn can be used by the end users via the cloud communications without any installation. Moreover, the end users' data files can be accessed and manipulated from any other computer using the internet services. Despite the flexibility of data and application accessing and usage that cloud computing environments provide, there are many questions still coming up on how to gain a trusted environment that protect data and applications in clouds from hackers and intruders.

Cloud storage should be able to store and share data securely, efficiently, and flexibly with others in cloud storage. The costs and complexities involved generally increase with the number of the decryption keys to be shared. The encryption key and decryption key are different in public key encryption. Since we are proposing new era of Aggregate key cryptography. To produce constant length ciphertext is also one of important task that we have materialized. In this paper, we propose a simple, efficient, and publicly verifiable approach to ensure cloud data security while sharing between different users. Since we introduce here, aggregate- key cryptosystem. Cryptographic methods are usually applied to address this data sharing issue.

© 2015, IJCSMC All Rights Reserved

I. Introduction:

Cloud computing is considered as the next step in the evolution of on-demand information technology which combines a set of existing and new techniques from research areas such as service-oriented architectures (SOA) and virtualization. With the rapid development of versatile cloud computing technology and services, it is routine for users to leverage cloud storage services to share data with others in a friend circle, e.g., Dropbox, Google Drive and AliCloud.

The shared data in cloud servers, however, usually contains users' sensitive information such as personal profile, financial data, health records, etc. and needs to be well protected. As the ownership of the data is separated from the administration of them, the cloud servers may migrate users' data to other cloud servers in outsourcing or share them in cloud searching. Therefore, it becomes a big challenge to protect the privacy of those shared data in cloud, especially in cross-cloud and big data environment. In order to meet this challenge, it is necessary to design a comprehensive solution to support user-defined authorization period and to provide fine-grained access control during this period [16][17].

Data cryptography mainly is the scrambling of the content of the data, such as text, image, audio, video and so forth to make the data unreadable, invisible or meaningless during transmission or storage is termed Encryption. And main aim of cryptography is to take care of data secure from invaders. The opposite process of getting back the original data from encrypted data is Decryption, which restores the original data. To encrypt data at cloud storage both symmetric-key and asymmetric-key algorithms can be used. But its having serious issues while handling huge database and transaction.

© 2015, IJCSMC All Rights Reserved

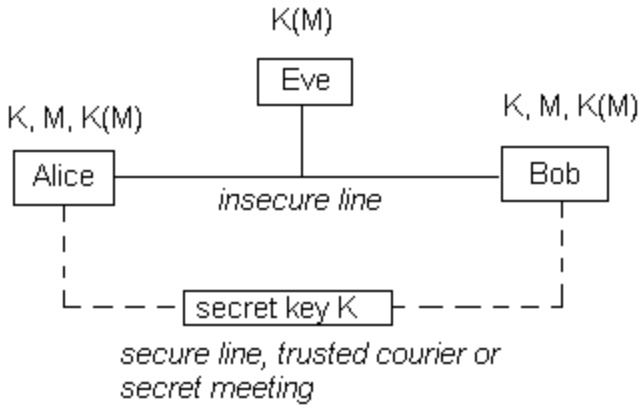

Fig 1: Public-key cryptography

Public-key cryptography, also known as asymmetric cryptography, is a class of cryptographic protocols based on algorithms that require two separate keys, one of which is secret (or private) and one of which is public. Although different, the two parts of this key pair are mathematically linked. The public key is used, for example, to encrypt plaintext or to verify a digital signature; whereas the private key is used for the opposite operation, in these examples to decrypt ciphertext or to create a digital signature.

1. The costs and complexities involved generally increase with the number of the decryption keys to be shared.
2. The encryption key and decryption key are different in public key encryption.
3. 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 owner's anonymity

II. RELATED WORK

Cloud computing is an internet base environment where users can store the data remotely in the cloud. Any cloud computing environment architecture can be divided basically into three layers, the characteristics layer, the models layer (infrastructure as a services, platform as a services, and software as a services), and the deployment layer [1].

These layers aim to (i) develop and adopt the rapidly evolving of cloud technology, (ii) abstract the details of inner implementations, and, (iii) facilitate the information retrieving service anywhere, anytime[2]. The following subsections explain the Cloud Data Encryption

© 2015, IJCSMC All Rights Reserved

Based Quantum (CDEQ) model and the Cloud Encryption Model (CEM) in details as they are the most popular models used in the encryption process based clouds.

A. Cloud Data Encryption Based Quantum (CDEQ)

Cloud data encryption based quantum technology platform dispels all security fears through cloud data transmission [3], [4]. This technology offers: simple low-cost data protection, tools and security services integration, and an efficient disasters recovery. Quantum technology solves one of the key challenges in distributed computing. It can preserve data privacy when users interact with remote computing centers [6]. Its power came from the deployment of the Quantum Cryptography or Quantum Key Distribution (QKD) mechanisms, which are considered as the art of the encryption/ encryption process [7], [8], see fig.1. Through quantum channels, data is encoded based on prepared states known as photons. These photons are then sent as "keys" for encryption/ decryption secured messages [9]. The advantage of using such photons in data transmission lays in the no-cloning theorem (the quantum state of a single photon cannot be copied).

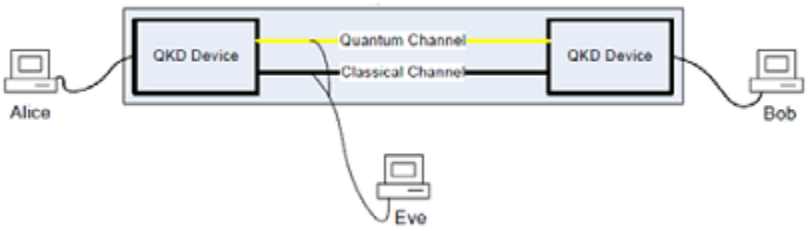

Fig. 2: Schematic of QKD

Bowfins looking for the perfect alliance between cloud computing and the quantum computing, which guarantees data protection for hosted files on remote computers or servers. He encrypted heavy duty of data by using the data processing servers as quantum computer, which succeeds in hiding input, processing and output data from malicious and attacks [10].

B. Cloud Encryption Models(CEM)

The two most important fields of information security in cloud environment are encryption and authentication. Generally, the encryption mechanism has become one of the basic priorities in maintaining the data security in the cloud, two popularity models based on data encryption technique are going to be explained briefly.

© 2015, IJCSMC All Rights Reserved

a) Cipher Cloud

Cipher Cloud provides a unified cloud encryption gateway with award-winning technology to encrypt sensitive data in real time before it‘s sent to the cloud. It also protects enterprise data by using operations-preserving encryption and tokenization in both private and public cloud communication without affecting functionality, usability, or performance [10]. Cipher cloud provides ability to create a unified data protection policy across all clouds that users probably used to store data, such as Google, Amazon, Azure and others. [11]. One the cipher cloud advantages are the offering of multiple AES-compatible encryption and tokenization options, including format and function-preserving encryption algorithms. Users see the real data when accessing an application through the Cipher Cloud security gateway, whereas the data stored in a cloud application is encrypted [12].

By applying encryption in a cloud security gateway, Cipher Cloud eliminates the inherent security, privacy, and regulatory compliance risks of cloud computing [13].

Cipher Cloud‘s highly secured encryption preserves both the format and function of the data, so that cloud applications remain operational, but their real content remains locked within the enterprise [13]. After then, the process is reversed when employees access cloud applications through the appliance decrypting data in real time so that users see the actual data rather than the encrypted version that resides within the cloud.

b) Cryptographic Cloud Storage

Kamara and Lauter et al [14] proposed a virtual private storage services that would satisfy the standard demands (Confidentiality, integrity, Authentication .etc.). Most of the demands are done by encrypting the documents stored in the cloud. However, such encryption leads to hardness in both the search processes through documents and the collaboration process in real time editing.

Fig. 3 shows the architecture of the cryptographic storage service that are used in solving the security problems of “back-ups, archival, health record systems, secure data exchange and e-discovery” [14]. It contains three main components: Data Processor (DP) that processes data before sending it to the cloud, Data Verifier (DV) which verifies data’s integrity and finally, Token Generator (TG) that generates tokens allowing the service provider to retrieve documents.

Before uploading data to the cloud, Aliceuses the data processor to encrypt and encode the documents along with their metadata (tags, time, size, etc.), then she sends them into the

© 2015, IJCSMC All Rights Reserved

cloud. When she wants to download some documents, Alice uses the TG to generate a token and a decryption key.

The token is sent to the storage provider to select the encrypted files to be downloaded. After that, the DV is invoked to verify the integrity of the data using a master key. The document is decrypted using the decryption key [14].

III. Theoretical Foundation:

Aggregation of Secret Keys:

Introducing a special type of public-key encryption which we call key-aggregate cryptosystem (KAC).I n KAC, users encrypt a message not only under a public key , but also under an identifier of ciphertext called class. The key owner holds a master-secret called master-secret key, which can be used to extract secret keys for different classes. More importantly, the extract key can be an aggregate key which is compact as a secret key for a single class, but aggregates the power of many such keys, i.e., the decryption power for any subset of ciphertext classes[15].

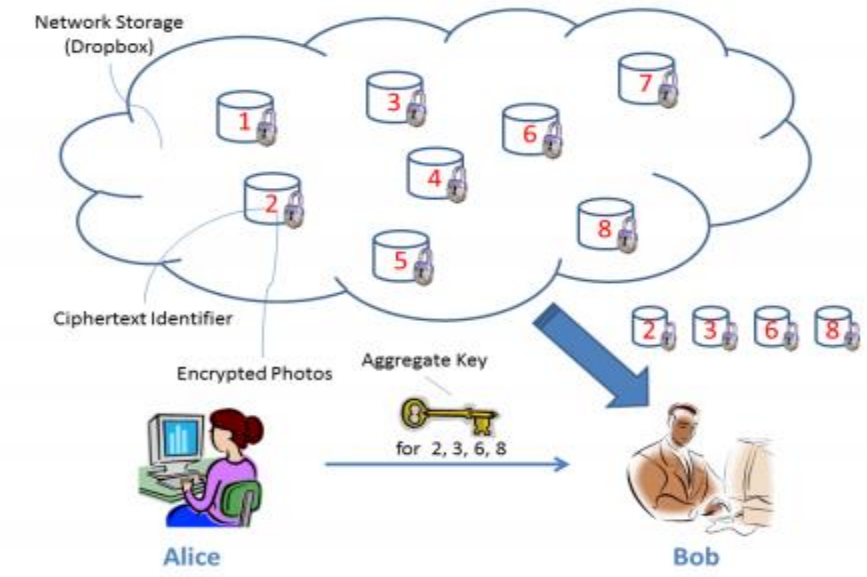

Fig 3: Alice share few files with Bob

Fig 3. Alice shares files with identifiers 2, 3, 6 and 8 with Bob by sending him a single aggregate key. Since our objective of our developing system is to design an efficient public-key encryption scheme which supports flexible delegation in the sense that any subset of the

© 2015, IJCSMC All Rights Reserved

ciphertexts is ecryptable by a constant-size decryption key (generated by the owner of the master-secret key)."

IV. **System Architecture**:

Here we describe the main idea of data sharing in cloud storage using KAC, illustrated in following figure.

Suppose Alice wants to share her data m1,m2,….m_n on the server. She first performs Setup (1^{λ}; n) to get param and execute KeyGen to get the public/master-secret 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 Alice. Anyone (including Alice herself) can then encrypt each m_i by C_i = Encrypt (pk, I, m_i). The encrypted data are uploaded to the server.

With param and pk, people who cooperate with Alice can update Alice's data on the server. Once Alice is willing to share a set S of her data with a friend Bob, she can compute the aggregate key K_S for Bob by performing Extract (msk, S). Since K_S is just a constant size key, it is easy to be sent to Bob via a secure e-mail.

After obtaining the aggregate key, Bob can download the data he is authorized to access. That is, for each i → S, Bob downloads C_i (and some needed values in param) from the server. With the aggregate key K_S, Bob can decrypt each C_i by Decrypt(K_S, S, I, C_i) for each i → S.

© 2015, IJCSMC All Rights Reserved

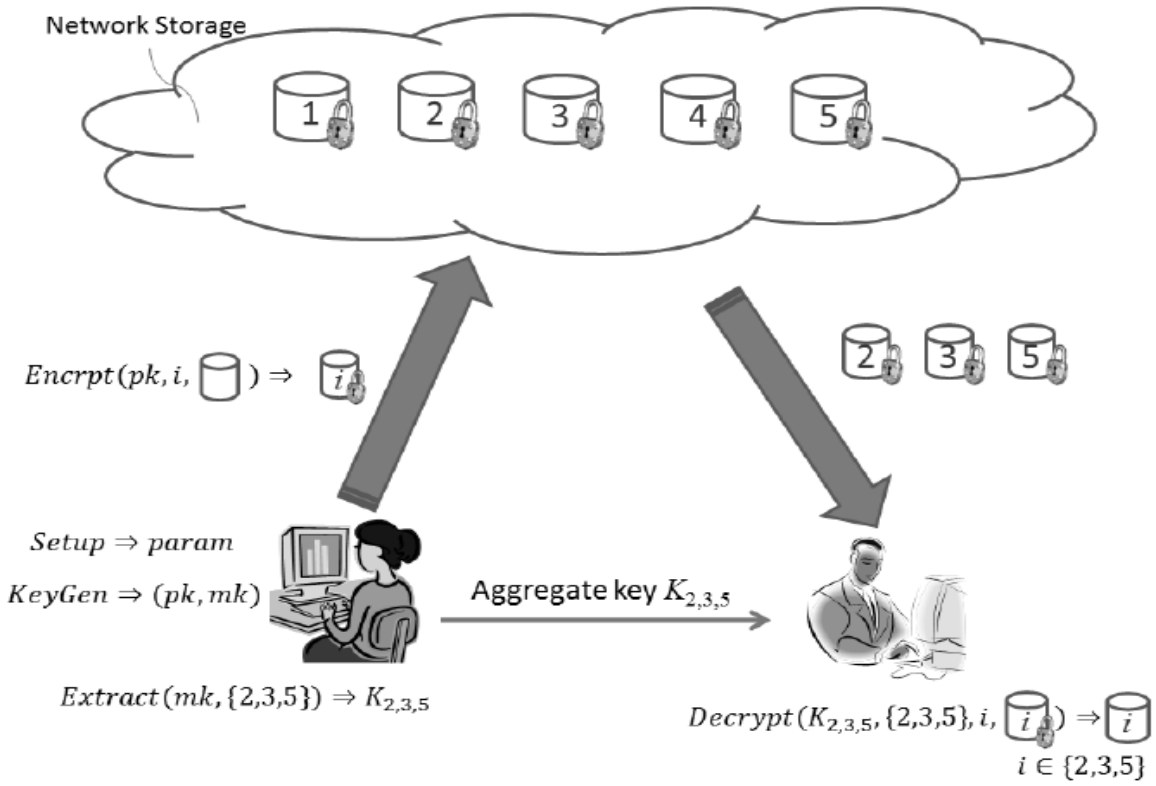

Fig 4. System Architecture

V. System Implementation:

a) Problem Statement:

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

Alice then uploads and shares those documents with the receiver 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

© 2015, IJCSMC All Rights Reserved

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.

We solve this problem by introducing a special type of public-key encryption which we call key-aggregate cryptosystem (KAC). In KAC, users encrypt a message not only under a public-key, but also under an identifier of ciphertext called class. That means the ciphertexts are further categorized into different classes. The key owner holds a master-secret called master-secret key, which can be used to extract secret keys for different classes. More importantly, the extracted key have can be an aggregate key which is as compact as a secret key for a single class, but aggregates the power of many such keys, i.e., the decryption power for any subset of ciphertext classes.

b) Key Aggregate framework:

The proposed system is basically design on the basis of key aggregation encryption. Here we are using two keys to encrypt and decrypt the data which are secret key and its aggregate key. The data owner creates the public system parameter and generates a secrete key which is public key. Data can be encrypted by any user and he may decides ciphertext block associated with the plaintext file which want to be encrypted.

The data owner have rights to use the secret key from which he can generate an aggregate key which is use for decryption of a set of ciphertext blocks. The both keys can be sent to end user in very secure manner. The authenticated user having an aggregate key can decrypt any block of ciphertext. This project consists of five algorithms which are used to perform the above operations.

These algorithms implementations having following steps are as follow:

Step1. Setup and create the account on the server for sharing of data. This account is generated by data owner.

Step2. KeyGen algorithm is used for the generation of public key. The data owner generates a public secrete key to encrypt the data over cloud. It also creates an aggregate key to access the block of ciphers of limited size.

Step3. Encrypts the data provided by the data owner by using the secrete key. This encrypted data is then share among the cloud.

© 2015, IJCSMC All Rights Reserved

Step4. The aggregate key is used for extracting the particular block of the ciphers from the cipher file. But other encrypted data remains secure.

Step5. Decrypt: The encrypted data is then decrypted by using the same secrete key which is use for encryption.

c) Aggregation of Secret Keys:

Introducing a special type of public-key encryption which we call key-aggregate cryptosystem (KAC). In KAC, users encrypt a message not only under a public key, but also under an identifier of ciphertext called class. The key owner holds a master-secret called master-secret key, which can be used to extract secret keys for different classes.

More importantly, the extract key can be an aggregate key which is compact as a secret key for a single class, but aggregates the power of many such keys, i.e., the decryption power for any subset of ciphertext classes.

Framework:

Following are different framework activity perform for executing KAC encryption;

Step1: The data owner establishes the public system parameter via Setup.

Step2: It generates a public/ master-secret key pair via KeyGen.

Step3: Messages is encrypted via Encrypt .

Step4: Cipher text class is associated with the plaintext message which is to be encrypted.

Step5: The data owner uses the master-secret to generate an aggregate decryption key for a set of ciphertext classes via Extract.

Step6: The generated keys is passed to delegates securely through secure e-mails or secure channels.

Step7: Receiver with an aggregate key decrypts any ciphertext provided that the ciphertext's class is contained in the aggregate key.

© 2015, IJCSMC All Rights Reserved

VI. System Mathematical Modeling

The proposed system S is defined as follows:

S = { I, O, F, U }

Where,
I: Input
O: Output
F: Functions
U: User

I= {U, IF, AU, ISM, DH}

Where

U = User which require Data Security in Cloud Using Key Aggregate Cryptosystem.

IF = Input files on cloud for sharing with others.
AU = Authenticated user login in to cloud by proving proper user id and password.
ISM = Input secret messages which system explicitly wants to share with other instead of sharing all files.
DH: Diffie Hellman algorithm input value for Key Exchange on cloud to share files on cloud like prime number etc…

O = {AK, DHSK, SF, CT, RFM}
Where below are the output generated from system processing;

AK = Key Aggregate Cryptosystem generates Aggregate Key for sharing files and Data Security in Cloud.
DHSK = Diffie Hellman secret key for exchanging aggregate key on cloud.
SF = Sharing of files on cloud with help of class identification fuction.
CT = System generate cipher text.
RFM = Finally receiver will receive files and display Received File Message.

U = {SD, FS, A, CA}
Where

SV = System developer
FS = Files Sender
FR = Files Receiver
A= Administrator
CA = Cloud Administrator

© 2015, IJCSMC All Rights Reserved

F= {F1, F2, F3, F4, F5}

Where

Function F1: To store the files on cloud for exchanging or sharing with other by providing data security.

Function F2: Sender sends files toward receiver by ecrypting with the help of key-aggregate cryptosystem (KAC) .

Function F3: To generate aggregate key of any set of secret keys and make them as compact as a single key.

Function F4: It uses Diffie-Hellman algorithm for exchanging secret key to ensures the confidentiality of the data on the cloud by using symmetric encryption.

Function F5: Receiver receives data files (ciphertext) and decrypts it only those having class identification tags.

- **Aggregate Key generation:**

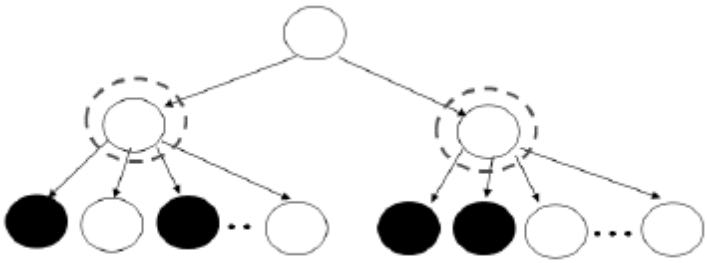

Fig.5: Key assignment in Key Aggregate Cryptosystem

Consider,

h: The height of the binary tree: there are total 2h ciphertext classes,

$\mathbf{n}_a$: The number of keys to be assigned,

N: The total number of keys in the hierarchy,

r: The delegation ratio: the ratio of the delegated ciphertext classes to the total classes.

If a user needs to classify his ciphertexts into more than n classes, he can register for additional key pairs (pk_2, msk_2), …. (pk_l, msk_l). Each class now is indexed by a 2-level index in {(i, j) | <= i<=*l*, 1<= j<= n} and the number of classes is increased by n for each added key [15].

© 2015, IJCSMC All Rights Reserved

- **Diffie-Hellman Algorithm for Key Exchange mathematical framework:**

Diffie–Hellman establishes a shared secret that can be used for secret communications while exchanging data over a public network. It exchange (D–H) is a specific method of securely exchanging cryptographic keys over a public channel.

For Example- The simplest and the original implementation of this protocol uses the multiplicative group of integers modulo *p*, where *p* is prime, and *g* is a primitive root modulo *p*. Here is an example of the protocol, with non-secret values in blue, and secret values in red.

1. Alice and Bob agree to use a prime number $p = 23$ and base $g = 5$ (which is a primitive root modulo 23).
2. Alice chooses a secret integer $a = 6$, then sends Bob $A = g^a \bmod p$
   - $A = 5^6 \bmod 23 = 8$
3. Bob chooses a secret integer $b = 15$, then sends Alice $B = g^{\boldsymbol{b}} \bmod p$
   - $B = 5^{\mathbf{15}} \bmod 23 = 19$
4. Alice computes $\boldsymbol{s} = B^{\boldsymbol{a}} \bmod p$
   - $\boldsymbol{s} = 19^{\mathbf{6}} \bmod 23 = 2$
5. Bob computes $\boldsymbol{s} = A^{\boldsymbol{b}} \bmod p$
   - $\boldsymbol{s} = 8^{\mathbf{15}} \bmod 23 = 2$

Alice and Bob now share a secret (the number 2).

VII. Case Study with Result Analysis:

Following Figure shows that Data Security in Cloud Using Key Aggregate Cryptosystem using the developed technique generates much less load on system about secret key as cyphertext classes increases.

In order to show the effectiveness of the developed method over the conventional and state-of-the-art Key Aggregate Cryptosystem techniques, several example of different area with different features are used.

We have checked our system on several input documents which are belongs to different category like Cryptographic Keys for a Predefined Hierarchy, Attribute-based encryption, Cloud Encryption Models, Compact Key in Identity-Based Encryption, Compact Key in Symmetric-

© 2015, IJCSMC All Rights Reserved

Key Encryption, etc. And every time our developed system has proved superiority among all existing systems. Here we are showing few snapshots of Key Aggregate Cryptosystem.

Figure 6: Front screen of KAC

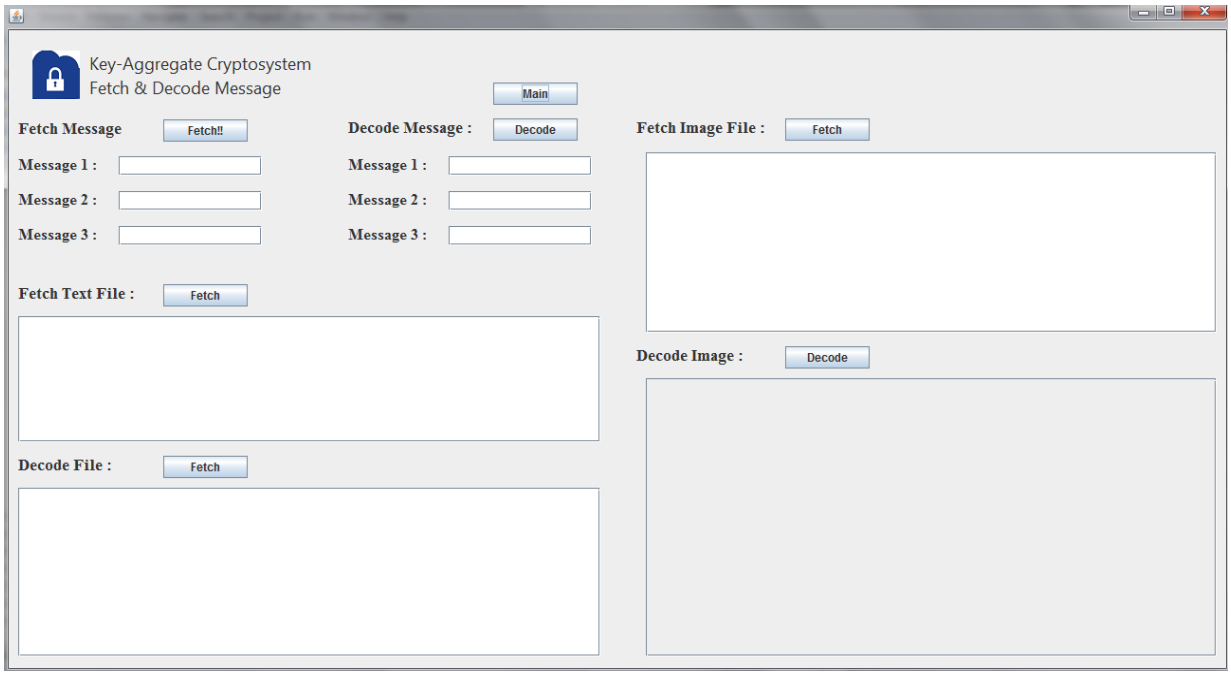

Figure 7: Above figure shows how to fetch file and decode and create aggregate key for sharing it on cloud.

© 2015, IJCSMC All Rights Reserved

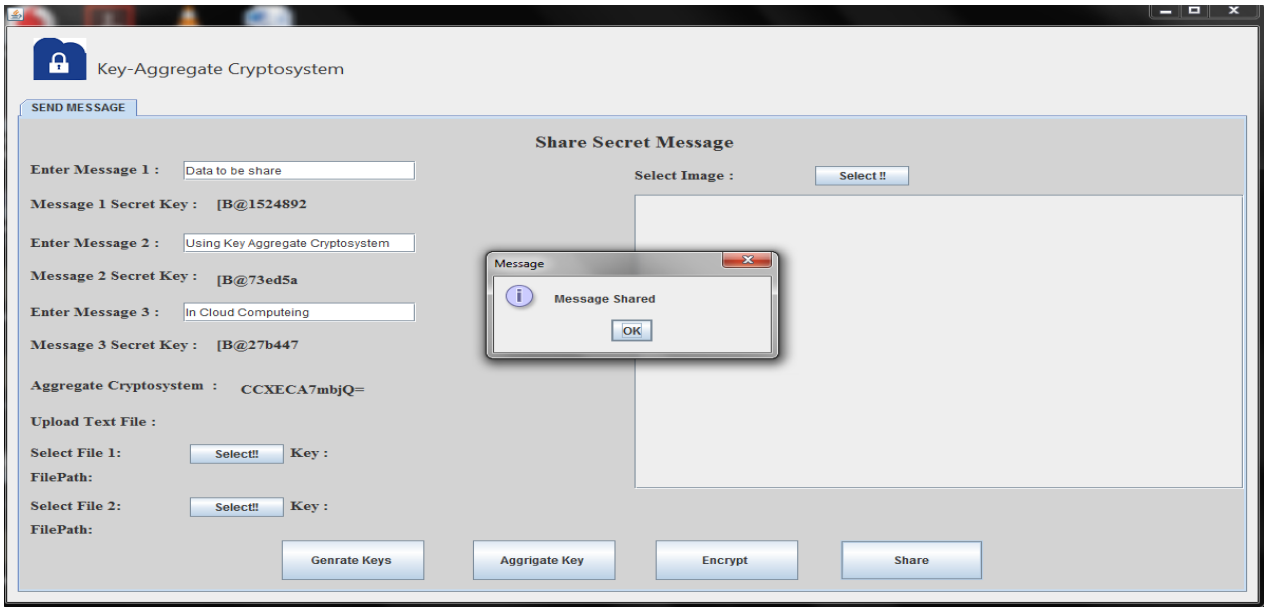

Fig 8: Sharing files on cloud using KAC and D-H Algorithm.

We can see that if we grant the key one by one, the number of granted keys would be equal to the number of the delegated ciphertext classes. With the tree-based structure, we can save a number of granted keys according to the delegation ratio. On the contrary, in our developed approach, the delegation of decryption can be efficiently implemented with the aggregate key, which is only of fixed size. Our approach is more flexible than hierarchical key assignment which can only save spaces if all key-holders share a similar set of privileges.

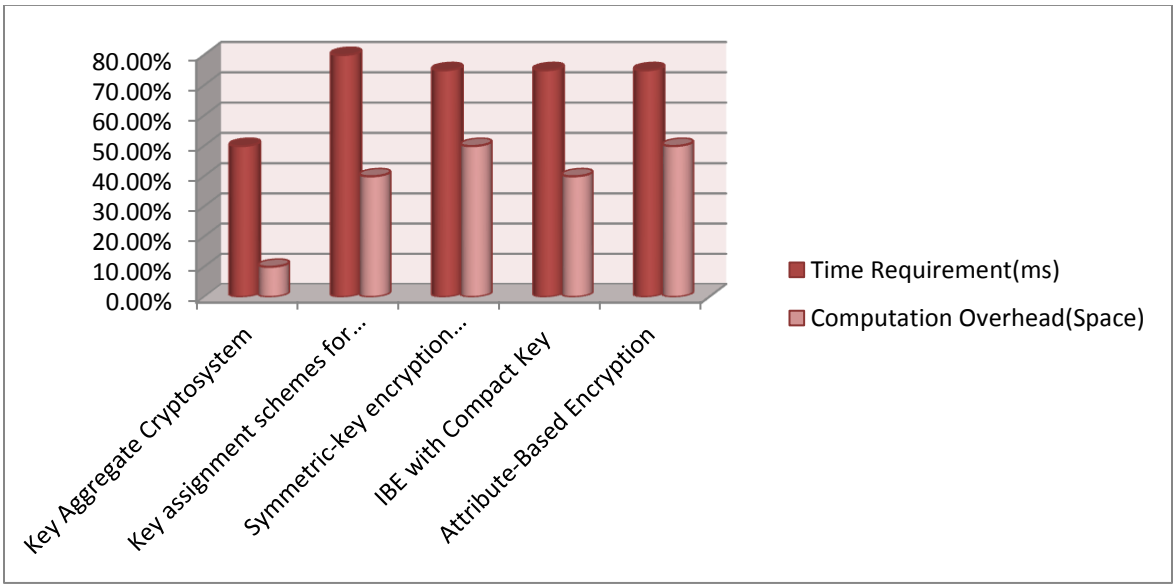

Figure 8: Comparison of Time requirement and Computational overhead in existing and developed system for providing data security.

© 2015, IJCSMC All Rights Reserved

VIII. Conclusion:

In this paper, we addressed an important issue of secure data sharing on untrusted storage. We investigated the challenges pertained to this problem and proposed data security in cloud using key aggregate cryptosystem.

In this paper, proposed system is found to be very efficient for sharing the data on cloud. For this we have used Key aggregate encryption algorithm which support delegation of secret keys for different ciphertext classes in cloud storage. It also produces constant-size ciphertexts such that efficient delegation of decryption rights for any set of ciphertexts which is here possible. Since in traditional methods unexpected privilege escalation will expose all data. And that we are able to avoid and provide more security by using key aggregate algorithm.

Graph also shows that requirement of time and space for computing this task on cloud is very less as compare with existing technology. It shows superiority of developed system than existing systems.

Acknowledgement:

I am very thankful to the people those who have provided me continuous encouragement and support to all the stages and ideas visualize. I am very much grateful to the entire VACOE for giving me all facilities and work environment which enable me to complete my task. I express my sincere thanks to PG Coordinator and my project guide Prof. M.K.Kshirsagar Sir, Head of the Computer Department Prof. Sarika Joshi, VA College of Engineering, Ahmednagar who gave me their valuable and rich guidance and help in presentation of this research paper.

References:

[1] G. Clarke, Microsoft's Azure Cloud Suffers First Crash, The Register, March 16, 2009, http://www.theregister.co.uk/ 2009/03/16/azure_cloud_crash/

[2] P. Ferrie, Attacks on Virtual Machine Emulators, White Paper, Symantec Corporation, January 2007, http://www.symantec.com/avcenter/reference/Virtual_Machine_Threats.pdf

[3] G. Fowler, B. Worthen, The Internet Industry is on a Cloud – Whatever That May Mean, The Wall Street Journal,March 26, 2010

© 2015, IJCSMC All Rights Reserved

[4] L. Youseff, M. Butrico, D. D. Silva, Toward a Unified Ontology of Cloud Computing, Grid Computing Environments Workshop, held with SC08, November 2008. http://www.cs.ucsb.edu/ ~lyouseff/ CCOntology/CloudOntology.pdf

[5] G. Ateniese, R. Di Pietro, L. V. Mancini, and G. Tsudik. Scalable and e_cient provable datapossession. In Proceedings of the 4th international conference on Security and privacy in communication networks (SecureComm '08), pages 1{10, New York, NY, USA, 2008. ACM.

[6] Hodges, A. (2005), 'Can quantum computing solve classically unsolvable problems"

[7] H.K. Lo, H.F. Chau, Unconditional security of quantum key distribution over arbitrary long distances. Science 1999; 283(5410): 2050-2056.

[8] L. Lydersen, Wiechers, C., Wittman, C., Elser, D., Skaar, J. and Makarov, V. Hacking commercial quantum cryptography systems by tailored bright illumination. Nat. Photonics 4, 686, 2010.

[9] K. Inoue, Quantum Key Distribution Technologies. IEEE Journal of Selected Topics in Quantum Electronics, vol. 12, no.4, July/August 2006.

[10] J. Brodkin. Gartner: Seven cloud-computing security risks. http://www.infoworld.com/d/security-central/gartnerseven- cloud computing- security-risks-853, 2008.

[11] C.C.A : CipherCloud Gateway Architecture, www.ciphercloud.net.

[12] M. v. Dijk and A. Juels. On the impossibility of cryptography alone for privacy-preserving cloud computing. In Hot topics in Security (HotSec'10), pages 1{8. USENIX Association, 2010.

[13] Kamara and Lauter . CS2: A Searchable Cryptographic Cloud Storage System,IJSIR,2012.

[14] G. Ateniese, S. Kamara, and J. Katz. Proofs of storage from homomorphic identi_cation protocols. In Advances in Cryptology - ASIACRYPT '09, volume 5912 of Lecture Notes in Computer Science, pages 319{333. Springer, 2012}.

[15] Cheng-Kang Chu, Sherman S. M. Chow, Wen-Guey Tzeng, Jianying Zhou "Key-Aggregate Cryptosystem for Scalable Data Sharing in Cloud Storage" IEEE Transactions on Parallel and Distributed Systems. Volume: 25, Issue: 2, 2014.

[16] 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.

© 2015, IJCSMC All Rights Reserved

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

[18] 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.

[19] 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.

[20] 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: LNCS, vol. 6805. Springer, 2012, pp. 442–464.

[21] 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.

© 2015, IJCSMC All Rights Reserved