© May 2016 | IJIRT | Volume 2 Issue 12 | ISSN: 2349-6002

SCALABLE DATA SHARING BY USING KEY AGGREGATION

Shivanand Gaikwad[1], Mukesh Giri[2], Akshay Jadhav[3], Rupesh kandhare[4], Prof.Nitin Hambir[5]
[1,2,3,4]*B.E.-Computer Engineering, Dr. D. Y. Patil School of Engineering, Pune, Maharashtra, India.*
[5]*Assistant Professor - Computer Engineering, Dr. D. Y. Patil School of Engineering, Pune, India.*

Abstract — **Cloud storage enables users to remotely store their data and enjoy the on-demand high quality cloud applications without the burden of local hardware and software management. Though the benefits are clear, such a service is also relinquishing users' physical possession of their outsourced data, which inevitably poses new security risks towards the correctness of the data in cloud. In order to address this new problem and further achieve a secure and dependable cloud storage service, key management and key sharing plays the main role in the data sharing concept of cloud computing. Aggregate key cryptosystem is a concept in which key is generated by means of various attributes of data in various cipher text classes and its associated keys. Aggregate key consist of various derivations of identity and attribute based classes of respective data owner of cloud. Irrespective of traditional cryptographic key generation and derivations, this technique possesses complex, unique and robust cryptographic key aggregate cryptosystem which is optimal for secure cloud and privacy preserving key generation process.**

I. INTRODUCTION

Cloud has become the best storage mechanism for all the users who access online resources and is gaining high popularity recently. Cloud storage plays a vital role in many personal applications as it the core technology for its existence. Many users are accessing the cloud space since Google Drive, One cloud etc. are providing access to the common user to make them aware about the convenience of the cloud storage and its access. When the wireless technology joined the hands with cloud it has turned to a miracle fulfilling every needs of the user from any corner of the world.

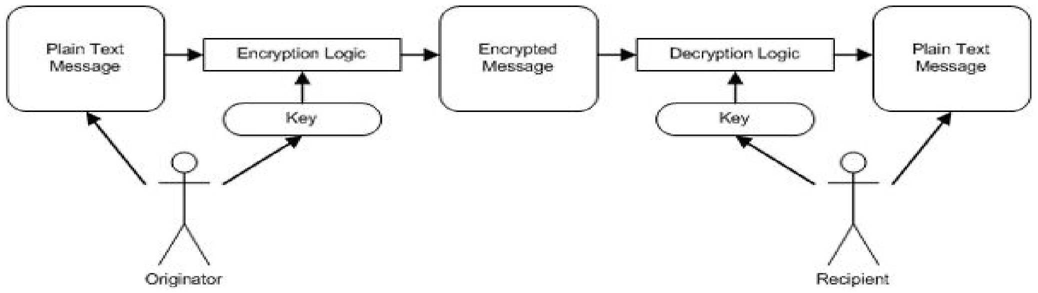
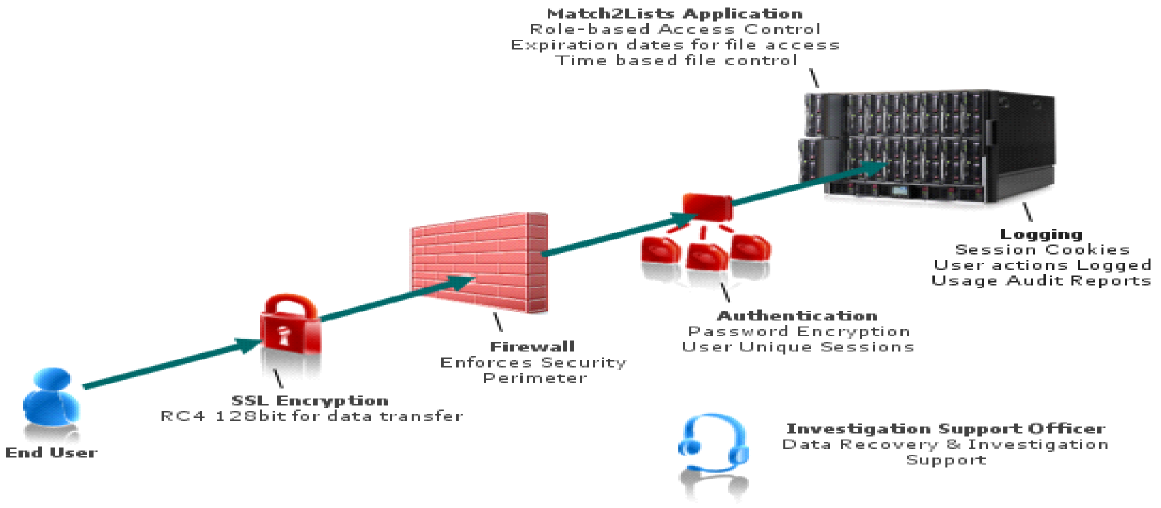


Fig Encryption and Decryption

The safekeeping of the data which is being amassed onto the cloud has become one of the key concerns. The trust of a cloud user cannot be relied blindly on a cloud provider completely. The confidentiality and integrity of the data cannot be assured if it is uploaded as such to the cloud. We depend on many cryptographic schemes to overcome this issue. Cryptographic schemes don't assure complete security but prevent the absolute revealing of the secret data. The major limitation arrives when the user needs to share the access to other on fine-grained level. One method is that the user has to provide the permission to access the complete data since the selected data permission can't be granted. Another method is that separate encryption has to be done the selected data one-by-one separately and send the private keys to the one who request. This is practically impossible when we consider the time, cost, complexity etc. Data can be so shared by encrypting all the selected data with its attributes and secret key converting it to a single aggregate key(private key) and this key can be sent over any communication channel like email, message etc. This mechanism not only saves the space, but also the execution time, cost, complexity etc.



Fig

Security Architecture of Cloud

The aggregate key can be used only to decrypt the data with which it was encrypted which means all the other data outside this set remains safe and hidden to the one to whom the aggregate key is being sent.

Our contribution: Cryptography is an amazing technique with which veils the veracity of the message from superfluous users. The Key-Aggregate Cryptosystem (KAC) [1] provides an outstanding performance reducing the computational complexity of the overall algorithm. The KAC aggregates various cipher texts into cipher text classes and every class holds a secret key from which the aggregate key will be generated. This generated aggregate key holds the decryption power of any subset of cipher classes

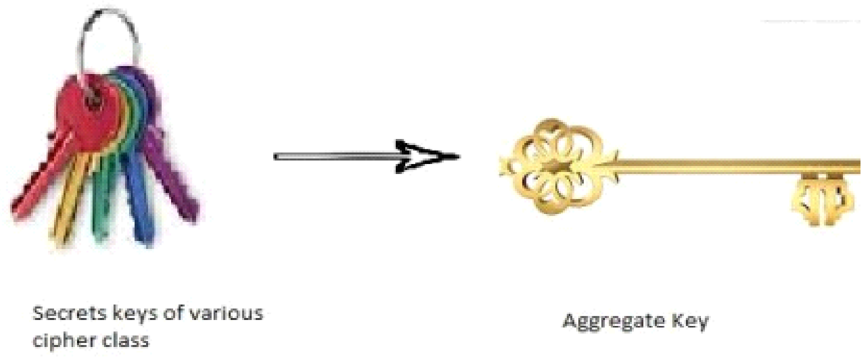


Fig: Multiple secret keys to single powerful Aggregate Key

Alice can send to bob the aggregate key as an email so that bob can decrypt the set of data which is being encrypted using the aggregate key and the set outside this encryption remain hidden to bob. Another advantage of this scheme is that the size of cipher text, aggregate key and the master secret key remains constant. KAC is a flexible work that the cipher text classes need not establish a relationship between each other [1].

We propose to perform the encryption and decryption process using the Diffie-Hellman since The Diffie-Hellman protocol is a method for two computer users to generate a shared private key with which they can then exchange information across an insecure channel. Let the users be named Alice and Bob. First, they agree on two prime numbers and , where is large (typically at least 512 bits) and is a primitive root modulo . (In practice, it is a good idea to choose such that is also prime.) The numbers and need not be kept secret from other users. Now Alice chooses a large random number as her private key and Bob similarly chooses a large number . Alice then computes , which she sends to Bob, and Bob computes , which he sends to Alice.

Now both Alice and Bob compute their shared key , which Alice computes as

and Bob computes as

Alice and Bob can now use their shared key to exchange information without worrying about other users obtaining this information. In order for a potential eavesdropper (Eve) to do so, she would first need to obtain knowing only , , and .
This can be done by computing from and from . This is the discrete logarithm problem, which is computationally infeasible for large . Computing the discrete logarithm of a number modulo takes roughly the same amount of time as factoring the product of two primes the same size as , which is what the security of the RSA cryptosystem relies on. Thus, the Diffie-Hellman protocol is roughly as secure as RSA.

II. KEY-AGGREGATE ENCRYPTION

The key-aggregate encryption process comprises of five polynomial-time algorithms as follows. [1] The data owner generates the public system parameter with the Setup algorithm and engenders a public/master-secret3 key pair through the KeyGen. Encryption of the messages to be stored on to the cloud can be done with the Encrypt algorithm. The master-secret key thus generated can be used to form the aggregate key in the Extract process. The generated aggregate key can be sent to delegate securely as an email or through portable devices. Finally, any client with an aggregate key can decrypt the data associated with this key receive though the process called Decrypt.

1. Setup: This is a randomized algorithm that takes no input other than the implicit security parameter.
2. KeyGen: randomly generate a public/mastersecret key pair (pk,msk).
3. Encrypt (pk,i,m): performed by anyone who is the owner of the data. Encrypts the data m using the public key and the index i of the cipher class and outputs C.
4. Extract (msk,S): this process results an aggregate key when we input the set of indices of the cipher class along with the master secret key.
5. Decrypt: decrypt is the process done by the one who receives the aggregate key obtaining the message m iff i ε S.

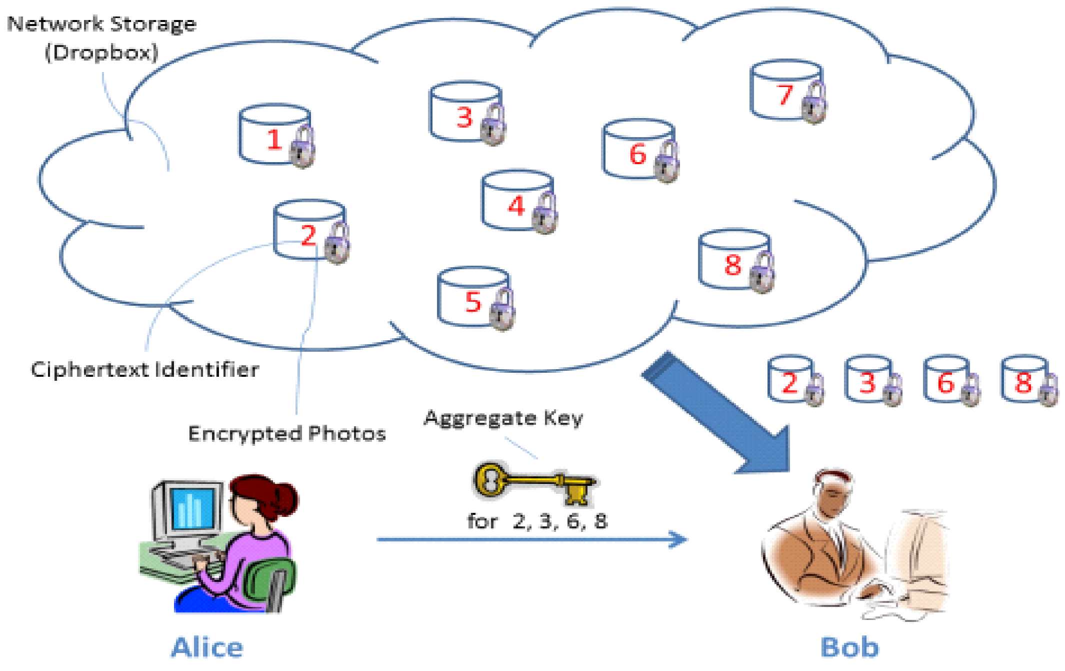


Fig 2.1 Data sharing in clouds

The above figure shows how the data is being shared in the cloud. Suppose a user Alice wants to share the data 1, 3, 5 to another user Bob, then the user Alice generates an aggregate key using the attributes of 1, 3, 5 and sends it to the user Bob. The user Bob thus decrypts the required data by performing the setup to generate the param, and then the keys are generated followed by the encryption process. The extract process generates the aggregate key with which the user Bob decrypts the data. In the traditional methods the key assignment will be providing separate keys for every data to be decrypted. This increases the key generation process as well as consumes much larger space.

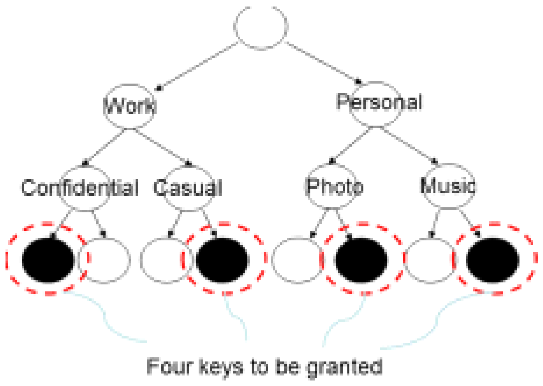


Fig. Key assignment for traditional cryptographic scheme

Here in the above figure four separate keys are to be granted for the availability of these files. The KAC uses only half the no. of keys than in the traditional cryptographic schemes. For example, consider the figure

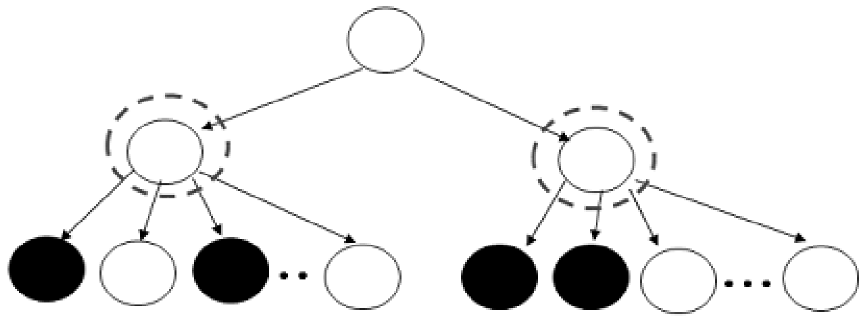


Fig 2.3 Key assignment in KAC

Only two different keys have to be provided for the access of five data. Hence hierarchical level of classification of data has a more advantageous level than the data arrangement in class level.

III. MATHEMATICAL EXPRESSION

Let P be the patient-controlled System; such that P = {I, F, O}
Where I is the set of inputs; I = {I1, I2, I3};
I1 =User information,
I2 = Patient information,
I3 =authorized data information;
F is a set of functions;
F = {F1, F2, F3, F4, F5, F6, F7, F8, F9, F10, F11, F12, F13, 14};
F1 = signup, F2 = sign-in, F3 = Patient details, F4 = view Patient information, F5 = add user, F6 = update information, F7 = delete information, F8 = schedule meetings, F9 = send invitations, F10 = register for user, F11 = information updates,
F12 = give feedback, F13 = key generations, F14 = budget analysis;
O is a set of outputs; O1 = {O1, O2, O3, O4, O5, O6, O7}; O1 = registration details, O2 = successful login, O3 = display event information, O4 = invitations sent successfully, O5 = registered for user, O6 = key generated, O7 = exit.

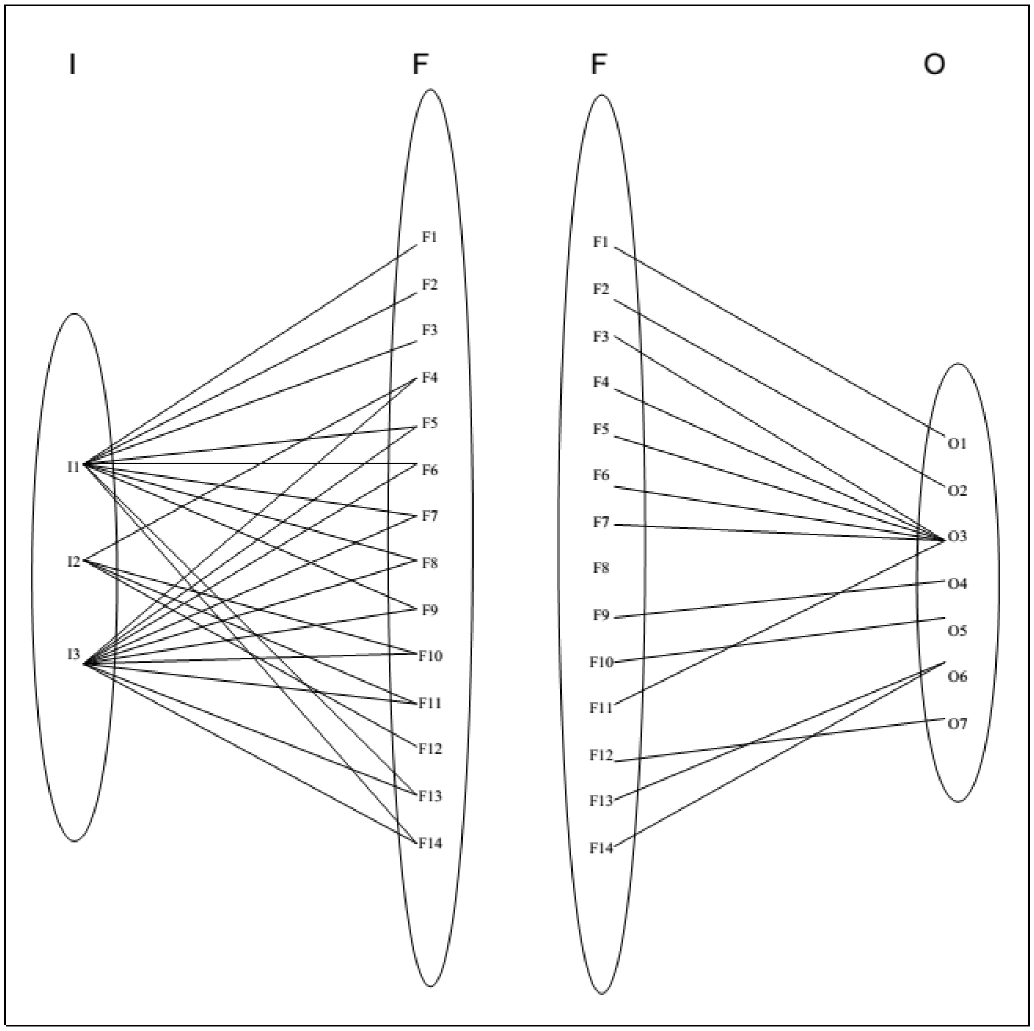


Fig. I/O Function Mapping

IV. RESULT

This has proposed a Scalable Data Sharing by using key aggregation is use to block any types of attack risk to cloud or data sharing on cloud and the most obvious purpose is provide single powerful Aggregate Key from multiple secret keys. No matter which one among the power set of classes, the delegate can always get an aggregate key of constant size.

V. CONCLUSION

In this paper, we have presented a single powerful Aggregate Key from multiple secret keys. No matter which one among the power set of classes, the delegate can always get an aggregate key of constant 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.

VI. FUTURE SCOPE

How to protect users' data privacy is a central question of cloud storage. With more mathematical tools, cryptographic schemes are getting more versatile and often involve multiple keys for a single application.

ACKNOWLEDGMENT

We would like to thank Dr. D. Y. Patil School of Engineering for providing us with all the required services. We are also grateful to Prof. Nitin Hambir and Prof. S. S. Das (Head of Computer Engineering Department), DYPSOE, Lohegaon, Pune for their vital support, suggestions and motivation during the entire course of the project.

REFERENCES

[1] Cheng-Kang Chu, Sherman S.M. Chow, Wen-Guey Tzeng, Jianying Zhou, and Robert H. Deng,” Key-Aggregate cryptosystem for Scalable Data Sharing in Cloud Storage,” IEEE Transactions on Parallel and Distributed Systems, vol. 25, no. 2, February 2014.

[2] Milind Mathur, Ayush Kesarwani, ”comparison between DES , 3DES , RC2 , RC6 , Blowfish and AES,” Proceedings of National Conference on New Horizons in IT - NCNHIT 2013.

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

[4] E. Horvitz, and K. Lauter, J., Benaloh, M. Chase ―Patient Controlled Encryption: Ensuring Privacy of Electronic Medical Records, in Processing of ACM Workshop on Cloud Computing Security (CCSW ’09). ACM, 2009, pp. 103–114.

[5] T. Okamoto and K. Takashima, ―Achieving Short Cipher texts or Short Secret-Keys for Adaptively Secure General Inner-Product Encryption, in Cryptology and Network Security (CANS ’11), 2011, pp. 138–159.