Volume 5, Issue 9, September 2015 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Secure Compressed Key Sharing for Multiple Cipher Text in Cloud Storage

[1]M. Vilasini, [2]Dr. K. Suresh Babu
[1](M.Tech), [2]Professor
[1, 2] Department of CSE, Vasireddy Venkatadri Institute of Technology (VVIT)
Andhra Pradesh, India

Abstract- ***We propose a perfect key aggregate system with scalable sharing of data in cloud. This scheme provides secure data storage and retrieval. Along with the security the access policy is also hidden for hiding the user's identity. This scheme is so powerful since we use aggregate encryption (AES) and improved Diffe Hellman key exchange technique in a single scheme. The novelty is that one can aggregate any set of secret keys and make them as compact as a single key, but concluding the power of all the keys being aggregated. The scheme detects any change made to the original file and if found clear the Error's. In order to improve the Security we used advanced cryptography technique for data encryption and decryption. So that by proposing those techniques we can provide more secure, efficient and flexible of sharing data. And also address the number of cipher texts usually grows rapidly without any restrictions. So we have to reserve enough cipher text classes in other hand we need to expand the public-key. Although the parameter can be downloaded with cipher texts, it would be better if its size is independent of the maximum number of cipher text classes.***

Keywords: Aggregate key cryptosystem, Cloud storage, data sharing, key-aggregate encryption. Diffe Hellman key exchange technique.

I. INTRODUCTION

Cloud storage is nowadays very popular storage system. Cloud storage is storing of data off-site to the physical storage which is maintained by third party. Cloud storage is saving of digital data in logical pool and physical storage spans multiple servers which are manage by third party. Third party is responsible for keeping data available and accessible and physical environment should be protected and running at all time. Instead of storing data to the hard drive or any other local storage, we save data to remote storage which is accessible from anywhere and anytime. It reduces efforts of carrying physical storage to everywhere. By using cloud storage we can access information from any computer through internet which omitted limitation of accessing information from same computer where it is stored. While considering data privacy, we cannot rely on traditional technique of authentication, because unexpected privilege escalation will expose all data. Solution is to encrypt data before uploading to the server with user's own key. Data sharing is again important functionality of cloud storage, because user can share data from anywhere and anytime to anyone. For example, organization may grant permission to access part of sensitive data to their employees. But challenging task is that how to share encrypted data. Traditional way is user can download the encrypted data from storage, decrypt that data and send it to share with others, but it loses the importance of cloud storage. Cryptography technique can be applied in a two major ways- one is symmetric key encryption and other is asymmetric key encryption.

In symmetric key Encryption, same keys are used for encryption and decryption. By This scheme is so powerful since they use aggregate encryption algorithms which are very simple so that large number of data can be stored in cloud without any problem in a single scheme. The scheme detects any change made to the original file and if found clear the errors. They explain public-key cryptosystems which produce a set of constant-size cipher texts such that efficient delegation of decryption rights for any set of cipher texts is possible. The best thing is that it is very easy to combine aggregate key into a single key, but encompassing the power of all the keys being aggregated. The data owner has the aggregate decryption key which is extracted from different ciphertext classes, but the other encrypted files outside the set remain confidential and very much authenticated. The advantage of this scheme is it provides secure data storage and retrieval and also the schemes detects any changes made to the original file stored in cloud and clear the errors if any changes found. The disadvantage of this scheme is it consumes more time for checking and recovery of every file.

A key-aggregate encryption scheme consists of five polynomial-time algorithms.

Setup The data owner executes the setup phase for an account on server which is not trusted. The setup algorithm only takes implicit security parameter.

KeyGen This phase is executed by data owner to generate the public or the master key pair (pk, msk).

Encrypt this phase is executed by anyone who wants to send the encrypted data. Encrypt (pk, m, i), the encryption algorithm AES takes input as public parameters pk, a message m, and i denoting ciphertext class. The algorithm encrypts message m and produces a ciphertext C.

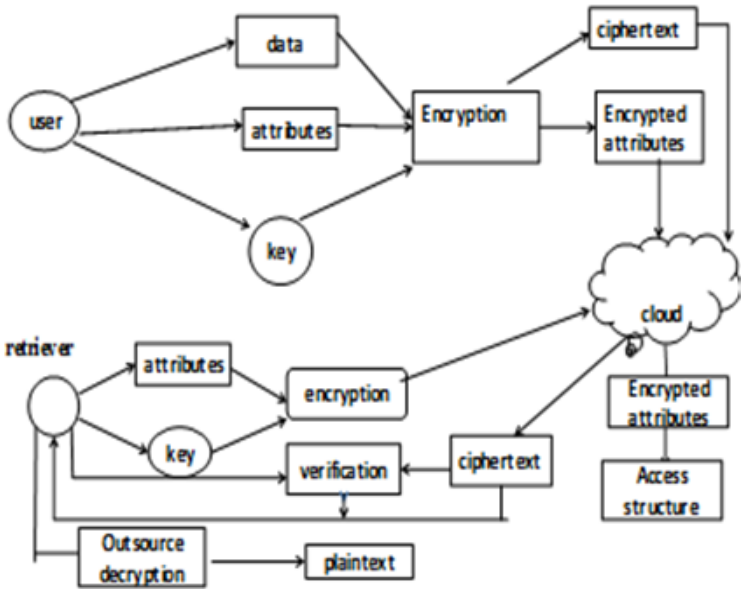


Fig 1 Secured data sharing architecture in cloud

Extract This is executed by the data owner for delegating the decryption power to the users by providing his Aggregate Decryption key.

Decrypt this is executed by the candidate who has the decryption authorities. Decrypt (kS, S, i, C), the decryption algorithm takes input as public parameters pk, a ciphertext C, i denoting ciphertext classes for a set S of attributes.

For example Alice wants to upload her data on the server. First she need to Setup an account on the server with security level parameter(1) and ciphertext classes(n) and then the public(pk) and master-secret key(msk) is generated by KeyGen algorithm. The data and index are encrypted by Alice as Encrypt (pk,i,m). If Bob wants to access her data on cloud he need to know an Aggregate key. Alice's master-secret key is used to compute the aggregate key by performing Extract (msk,S). Then Bob can be able to download the data from the server by Decrypt (Ks,S,i,Ci).

II. LITERATURE SURVEY

In 2006 V. Goyal, O. Pandey, A. Sahai, and B. Waters, worked on ―Attribute-Based Encryption for Fine-Grained Access Control of Encrypted data‖, this paper adds to another cryptosystem for fine-grained sharing of encrypted data. This plan was called Key-Strategy Property Based Encryption (KP-ABE). In our cryptosystem, figure writings are marked with sets of characteristics and private keys are connected with access structures that control which figure messages a client has the capacity unscramble [4]. Points of interest: - Relevance of KP-ABE plan is to sharing of review log information and show encryption

In 2007 F. Guo, Y. Mu, Z. Chen, and L. Xu,worked on ―Multi-Identity Single-Key Decryption without Random Oracles,‖ This Paper produce Multi-Character Single-Key Unscrambling (MISKD).It is a Personality Based Encryption (IBE) framework where a private decoding key can outline open keys (personalities). All the more precisely, in MISKD, a solitary private key can be utilized to unscramble various figure writings encrypted with distinctive open keys related to the private key [3]. Points of interest Multi-Character Single-Key Decoding plan is more productive in unscrambling.

In 2009 J. Benaloh, M. Chase, E. Horvitz, and K. Lauter, worked on ―Patient Controlled Encryption: Ensuring Privacy of Electronic Medical Records,‖ . This framework assemble an effective framework that permits patients both to impart fractional access rights to others, and to perform seeks over their records. We formalize the necessities of a Patient Controlled Encryption plan, and give a few occasions, taking into account existing cryptographic primitives and conventions, every accomplishing an alternate arrangement of properties [2].

Points of interest The patient can undoubtedly allow access to a class Comparably, specialists can include subcategories with self-assertive names, without help from the patient. This will be especially valuable on the off chance that we can't foresee the names of all conceivable subcategories, If a specialist needs to include a class for another kind of test, or if classifications are named by visit dates.

In 2009,M. J. Atallah, M. Blanton, N. Fazio, and K. B. Frikken, worked on ―Dynamic and Efficient Key Management for Access Hierarchies,‖. The proposed arrangement has the accompanying properties: (i) just hash capacities are utilized for a hub to get a relative's key from its own particular key; (ii) the space unpredictability of people in general information is the same as that of putting away the pecking order; (iii) the private information at a class comprises of a solitary key connected with that class; (iv) redesigns (repudiations, augmentations, and so forth.) are taken care of locally in the progressive system; (v) the plan is provably secure against plot; and (vi) key determination by a hub of its relative's key is limited by the quantity of bit operations direct in the length of the way between the nodes[1]. Favorable circumstances the dynamic plan accomplish a most noticeably bad and normal case number of bit operations for key determination that exponentially superior to the profundity of an adjusted pecking order.

III. SYSTEM OBJECTIVE

The Objective of the system is to provide best solution for the Existing problem is that Alice encrypts files with distinct public-keys, but only sends Bob a single (constant-size) decryption key while sharing the files cipher text class index also considerable when a same user shares multiple files class index remain constant i.e no variation in class index. Since the decryption key should be sent via a secure channel and kept secret, small key size is always desirable. Using the Public-key Cryptosystem (public key Encryption algorithm)

PROBLEM DEFINITION:-

The challenging problem is how to effectively same user share multiple encrypted files thus class index remains same (constant). Of course users can download the individual or bulk encrypted files 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 directly

SCOPE:-

We Can 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. In this paper, we consider how to "compress" secret keys in public-key cryptosystems which support delegation of secret keys for different cipher text classes for different user and a single cipher text class index remains constant for same user in cloud storage. No matter which one among the power set of classes, the delegate can always get an aggregate key of constant size.

PROBLEM TESTIMONIAL

Constant-size decryption key require pre-defined hierarchical relationship. The fixed hierarchy is used. In that there is only one way in which we can partition the record. If we want to give out access rights based on something else (e.g. based on document type or sensitivity of data) we will have to look at all the low-level categories involved, and give a separate decryption key for each [2]. More number of decryption key was used [1].

SYMMETRIC-KEY ENCRYPTION WITH COMPACT KEY AGGREGATE CRYPTOSYSTEM

The proposed system design an efficient public-key encryption scheme which supports flexible allocation. In this scheme any subset of the cipher texts (produced by the encryption scheme) is decrypt by a constant-size decryption key (generated by the proprietor of the master-secret key). We solve this problem by introducing a special type of public-key encryption called keyaggregate cryptosystem (KAC). In KAC, users encrypt a message not only under a public-key, but also under an identifier of cipher text called class. Such that cipher texts are further categorized into different classes. The owner of the key holds a master-secret called Master secret key [5].

The master-secret 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, such that the decryption power for any subset of cipher text classes. By this solution, Alice can simply send Bob a single aggregate key via a secure channel like email. Bob can download the encrypted photos from Alice's Drop box space and then use this aggregate key to decrypt these encrypted photographs.

IBE WITH COMPRESSED KEY

Identity-based encryption (IBE) (e.g., [5], [6], [7]) is a 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, mobile number). There is a private key generator (PKG) in IBE which holds a master-secret key and issues a secret key to each user with respect to the user identity. The content provider can take the public parameter and a user identity to encrypt a message. The recipient can decrypt this ciphertext by his secret key. Guo et al. [8], [9] tried to build IBE with key aggregation. 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 be aggregated.[1] This significantly increases the costs of storing and transmitting ciphertexts, which is impractical in many situations such as shared cloud storage. As Another way to do this is to apply hash function to the string denoting the class, and keep hashing repeatedly until a prime is obtained as the output of the hash function.[1] we mentioned, our schemes feature constant ciphertext size, and their security holds in the standard model. In fuzzy IBE [10], one single compact secret key can decrypt ciphertexts encrypted under many identities which are close in a certain metric space, but not for an arbitrary set of identities and therefore it does not match with our idea of key aggregation.

ATTRIBUTE-BASED ENCRYPTION

Attribute-based encryption (ABE) [11], [12] 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 $(1 \vee 3 \vee 6 \vee 8)$, one can decrypt ciphertext tagged with class 1, 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.g., [13]).

IV. PROPOSED STRUCTURE

The data owner establishes the public system parameter through Setup and generates a public/master-secret key pair through KeyGen. Data can be encrypted via Encrypt by anyone who also decides what ciphertext class is associated with the plaintext message to be encrypted. Here data owner can encrypt and share multiple files using same constant cipher text class index towards to reduce the no of Class index files for individual files thus it improve the performance and storage space. The data owner can use the master-secret key pair to generate an aggregate decryption key for a set of ciphertext classes through Extract. The generated keys can be passed to delegates securely through secure e-mails or secure devices Finally, any user with an aggregate key can decrypt any ciphertext provided that the ciphertext's class is contained in the aggregate key via Decrypt. Key aggregate encryption schemes consist of five polynomial time algorithms as follows:

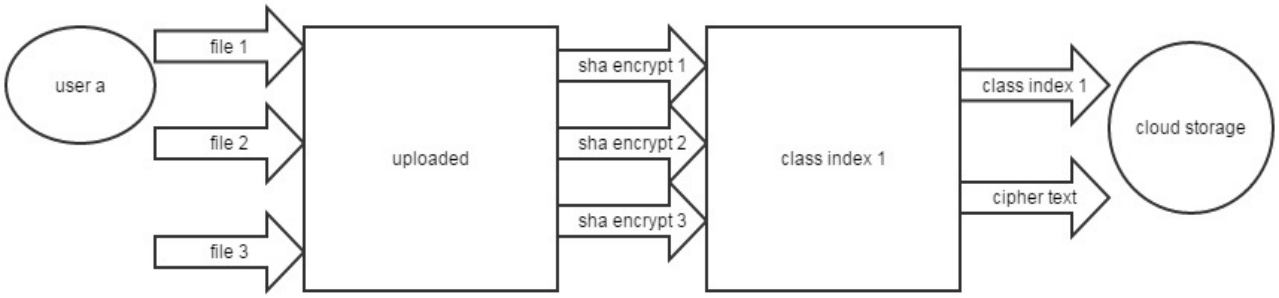


Fig 2. Proposed Architecture

1. SETUP (1λ , N) : The data owner establish public system parameter via Setup. On input of a security level parameter 1λ and number of ciphertext classes n , it outputs the public system parameter param

2. KEYGEN: It is executed by data owner to randomly generate a public/ master-secret key pair (Pk, msk).

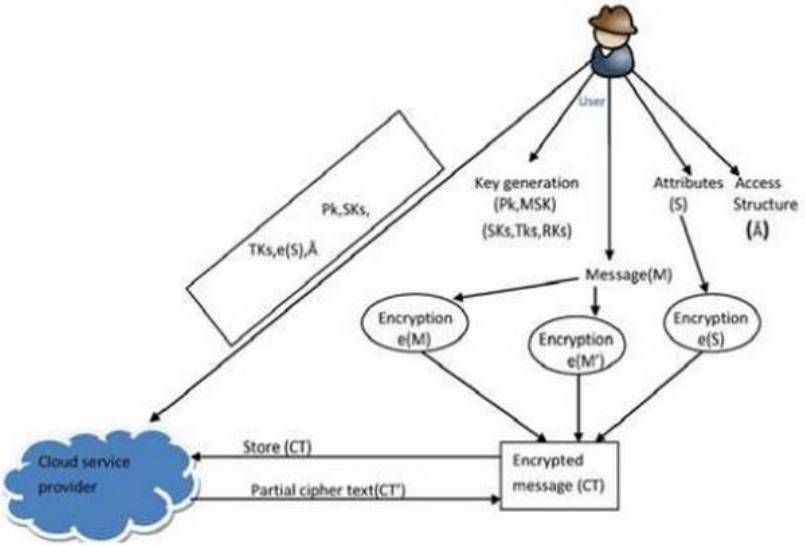


Fig 3. Key Generation Phase

3. ENCRYPT (PK, I, M) : It is executed by data owner and for message m and index i ,it computes the ciphertext as C.
For Encryption Advanced Encryption Standard AES is used , It is Symmetric algorithm with a block length of 128 bits. The key is arranged in the form of a matrix with 4 × 4 bytes. The key matrix is expanded into a schedule of 44 words. There are 10 rounds. For encryption, each round consists of the following four steps: 1) Substitute bytes, 2) Shift rows, 3) Mix columns, 4) Add round key Byte substitution (each value of the state is replaced with S-Box value) Shift rows (circular left shift on each row of the state) Mix columns (uses matrix multiplication in GF(256)) Add round key (bitwise XOR of current block with portion of expanded key) For decryption, each round consists of the following four steps: 1) Inverse shift rows, 2) Inverse substitute bytes, 3) Add round key, 4) Inverse mix columns.

4. EXTRACT (MSK, S): It is executed by data owner for delegating the decrypting power for a certain set of ciphertext classes and it outputs the aggregate key for set S denoted by Ks.

5. DECRYPT (KS, S, I, C): It is executed by a delegate who received, an aggregate key Ks generated by Extract. On input Ks, set S, an index i denoting the ciphertext class ciphertext C belongs to and output is decrypted result m

DATA SHARING

KAC in meant for the data sharing. The data owner can share the data in desired amount with confidentiality. KCA is easy and secure way to transfer the delegation authority.

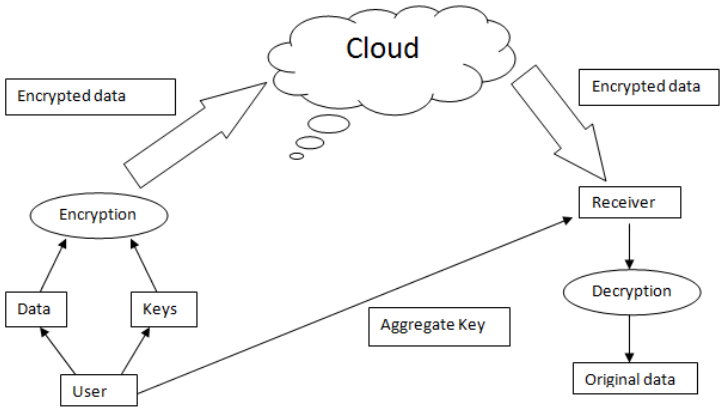


Fig 4. Data sharing architecture

For sharing selected data on the server Alice first performs the Setup. Later the public/master key pair (pk, mk) is generated by executing the KeyGen. The msk master key is kept secret and the public key pk and param are made public. Anyone can encrypt the data m and this data is uploaded on server. With the decrypting authority the other users can access those data. If Alice is wants to share a set S of her data with a friend Bob then she can perform the aggregate key KS for Bob by executing Extract (mk, S). As kS is a constant size key and the key can be shared through secure e-mail. When the aggregate key has got Bob can download the data and access it. .

For the key aggregate cryptosystem we are using advanced diffiehellman key exchange algorithm and advance XOR cryptography technique for the data encryption and decryption. The procedure of Advanced diffiehellman key exchange is as follows.

Shared key Generation process: In this section each user will generate secret key and send to KGC (key generation center). The KGC center will retrieve all secret key and compressed make as single secret key and send to each user. The procedure of advanced diffiehellman key exchange protocol as follows.

1. Each user select prime number p, g and private key a.
2. Using those values the user will calculate public key using pub=gamod p.
3. After calculating each user will send his public keys to KGC.
4. The KGC will retrieve public key from the each users and generate new private for the each users and calculate another public for individual users.
5. After generating each public of users and send to it.
6. Each user will retrieve public key coming from the KGC will generate shared key using shared key=gpub mod p.
7. After generating shared keys of each uses and send to KGC.
8. The KGC will retrieve all shared key from the users and XOR with those key to form one secret key.
9. After generating secret key the KGC will send to all users. Each user will use this secret key and encrypt shared data stored into cloud. If any user wants to retrieve that he can decrypt the data using secret key.

Advanced XOR Cryptography Technique:

In this section each user will encrypt and decrypt the shared data using this process. The process of advanced XOR cryptograph technique is as follows.

I)Encryption Process:

1. Generate the ASCII value of each character of Data.
2. After converting ASCII format we can XOR with secret key.
3. The resulted XOR data will convert into binary format i.e. each character length of should be 8 bits.
4. The 8 bit binary format of should be reversed.
5. We choose the four bit as divisor will treat as key (1000).
6. The reversed number can be devised by divisor and get first three digit as remainder and next five digits are quotient. If any digits will less than three or five digit the we add required number of zero in the left hand side. So that this is cipher text. This process will repeat until the total data can converted into cipher format. After converting cipher format that data can be stored into cloud. If any user will perform decryption process as follows.

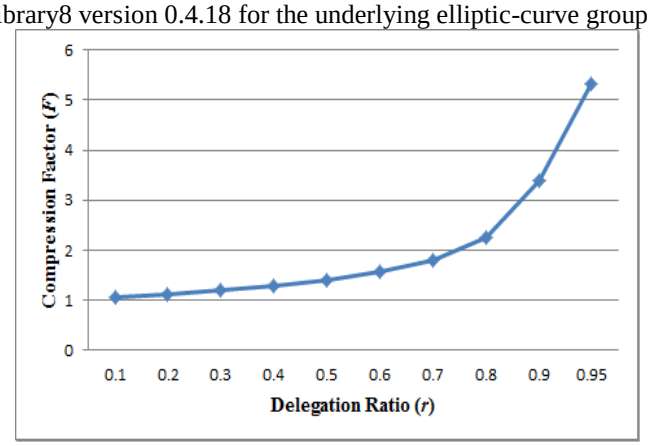
II) Decryption process:

1. Retrieve the cipher data and multiply last five digits with divisor.
2. The first three digits of cipher text with result produced by previous step.
3. The result of previous step of will not contain 8 bits that will make into 8 bit number.
4. Reverse resulted bit of given format and convert into ASCII format.
5. After converting into binary format we can XOR with key and result will be convert into character will get plain text until end of cipher data.

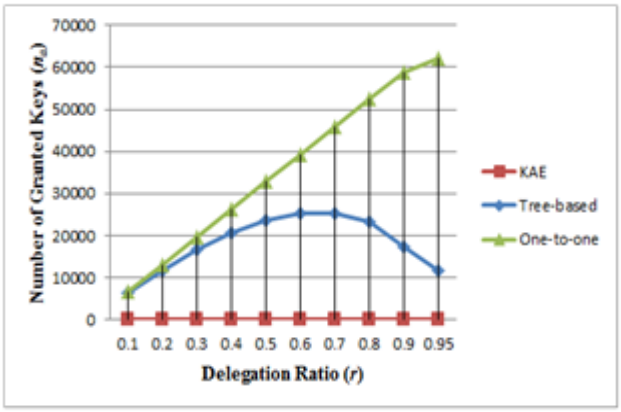
V. PERFORMANCE OF OUR PROPOSED SCHEMES

Our approaches allow the compression factor F (F =n in our schemes) to be a tunable parameter, at the cost of O(n)-sized system parameter. 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, where S is the set of ciphertext classes decryptable by the granted aggregate key and $|S| \leq n$. As expected, key extraction requires O(|S|) group multiplications as well, which seems unavoidable. However, as demonstrated by the experiment results, we do not need to set a very high n to have better compression than the tree-based approach.Note that group multiplication is a very fast operation.

Again, we confirm empirically that our analysis is true. We implemented the basic KAC system in C with the Pairing-Based Cryptography (PBC) Library8 version 0.4.18 for the underlying elliptic-curve group and pairing operations.



(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)

VI. CONCLUSION

In this paper outsourcing of data to cloud server through network with secure manner. For the privacy of data we are using so many cryptography technique are used. One of the techniques for key aggregate cryptosystem for generation of secret key with different set of secret keys. In this paper we are proposed concept of advanced diffie Hellman key exchange for generation of secret with multiple set of secret keys. Using this key the data owner or user will encrypt and decrypt stored data into cloud. By performing encryption and decryption of data we are using advanced XOR cryptography technique. By implementing those techniques we provide secrecy and confidentiality of data.

REFERENCES

[1] H.Fareesa Firdose , R.Deepthi Crestose Rebekah,"A Key Aggregate Construction with Adaptable Offering of Information in Cloud " INTERNATIONAL JOURNAL OF COMPUTER ENGINEERING IN RESEARCH TRENDS VOLUME 2, ISSUE 5, MAY 2015, PP 355-358 , ISSN (Online): 2349-7084. www.ijcert.org

[2] 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.

[3] S.Kamara and K.Lauter,—Cryptographic Cloud Storage,Proc.Int'l Conf. Financial Cryptography and Data Security (FC), pp. 136-149, Jan. 2010

[4] 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.

[5] V. Goyal, O. Pandey, A. Sahai, and B. Waters, Attribute-Based Encryption for Fine-Grained Access Control of Encrypted data, in Proceedings of the 13th ACM Conference on Computer and Communications Security (CCS '06). ACM, 2006, pp. 89–98.

[6] S. Yu, C. Wang, K. Ren, and W. Lou, Achieving Secure, Scalable, and Fine-Grained Data Access Control in Cloud Computing, Proc. IEEE INFOCOM, pp. 534-542, 2010.

[7] M. Chase and S. S. M. Chow, Improving Privacy and Security in Multi-Authority Attribute-Based Encryption, in ACM Conference on Computer and Communications Security, 2009, pp. 121–130

[8] 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.

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

[10] 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.

[11] V. Goyal, O. Pandey, A. Sahai, and B. Waters, "Attribute-Based Encryption for Fine-Grained Access Control of Encrypted data," in Proceedings of the 13th ACM Conference on Computer and Communications Security (CCS '06). ACM, 2006, pp.89–98.

