Achieving Constant-Size Aggregate Key for Efficient and Secured Data Sharing

Sakkubai M Horatti[1], Gopal B[2]

[1]M. Tech, Student, Department of computer science & Engineering, VTU University, MITE college, Moodabidri-574225, Mangalore.

[2]Assistant Professor, Department of computer science & engineering, VTU University, MITE college, Moodabidri-574225, Mangalore

Abstract: ***Cloud computing is a technology which maintains the data and applications using the internet and central remote servers. Cloud computing allows consumer and businesses to use applications without any installation by using the internet they will access their personal files on any computer. Cloud storage is a service model in which data are maintained, managed and backed up remotely and made available to users over a network. Cloud storage can provide better accessibility, and reliability, strong protection, disaster recovery and lower cost. In proposing method a new public–key encryption is introduced which is called as Key-aggregate cryptosystem (KAC). Key-aggregate cryptosystem turn out constant size ciphertexts specified economical delegation of decoding rights for any set of ciphertext area unit attainable. Any set of secret keys can be aggregated and make them as single key, which holds the power of all the keys being aggregated. This aggregate key can be sent to the others via secured channels for decryption of ciphertext set and remaining encrypted files outside the set are remains confidential.***

Keywords: Cloud computing, Cloud storage, Ciphertext, Decryption, Encryption, Key-aggregate cryptosystem, secret key.

1. Introduction

In the recent years, cloud storage is becoming overwhelmingly popular and considered a must have technology in the IT world. Many companies like Apple and Google are also making the cloud available to their customers for personal use. Cloud storage is saving of digital data in logical pool and physical storage spans multiple servers which are managed by third party. The 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 the same computer where it is stored.

Data privacy is a central question of cloud storage. While considering data privacy, we cannot rely on traditional techniques of authentication, because unexpected privilege escalation will expose all data [1]. The best solution for this problem is to encrypt the data before uploading to the server with user’s own key. The things become even worse in a shared lease cloud computing environment. Data from different users can be hosted on separate virtual machines (VMs) but resides on a single physical machine. Data in a target virtual machine could be stolen by instantiating another virtual machine occur together with the target one. There are a number of cryptographic schemes were proposed for ensuring the availability and security of files. This scheme allows a third-party auditor to check the availability of files on behalf of the data owner without leaking any information, or without compromising the data owner’s security. Similarly, cloud users will not hold the strong trust that the cloud server is doing a good quality job in terms of privacy. A cryptographic technique is the best solution for this problem. Whenever the user is not perfectly happy with trusting the security of the virtual machine or honesty of the technical staff. Those users are encrypting their data with their own keys before uploading the data 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 data; an enterprise may grant her employees access to a portion of sensitive data.

The one of challenging problem is how to effectively share encrypted data. Of course user can download the encrypted data from the cloud storage and decrypt them, then send them to others for sharing, but it loses the values 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. However, in cloud storage finding an efficient and secure way to share partial data is not trivial. Below we will take Dropbox as an example for illustration. Assume that Alice puts all her private photos on Dropbox, and she does not want to expose her photos to everyone. Due to various leakage possibilities Alice cannot feel relieved by just depending on the privacy protection mechanisms provided by Dropbox, so using her own keys she encrypt all the photos before uploading to server. One day, Alice’s friend, Bob, asks her to share the photos taken over all these years which Bob appeared in. Alice can then use the share functions of Dropbox, but the problem now is how to delegate the decryption rights for these photos to Bob. Alice has two possibilities under the traditional encryption pattern:

- Alice encrypts all files with a single encryption key and gives Bob the corresponding secret key directly.
- Alice encrypts each file with distinct keys and sends Bob the corresponding secret keys.

Obviously, the first method is lacking the quality since all un-chosen data may be leaked to Bob. For the second method, there are practical concerns on efficiency. The number of such keys is as much as the shared photos, say, a thousand. To transfer these secret keys inherently requires a secure channel, and storing these keys requires rather expensive secure storage. The costs and complexities involved generally increase with the number of the decryption keys to be shared. In short, it is very heavy and costly to do that.

1.1 Types of Encryption

Encryption keys also come in two flavors — symmetric key and asymmetric (public) key. In symmetric encryption only single key is used for both encryption and decryption, but in asymmetric encryption different keys are used for encryption and decryption.

- Symmetric Key Encryption

Using symmetric encryption, when Alice wants the data to be originated from a third party, she has to give the Encryptor her secret key. Obviously, this is not always desirable.

- Asymmetric Key Encryption

By contrast, the encryption key and decryption key are different in public-key encryption. The use of public-key encryption gives more flexibility for our applications. For example, in an enterprise setting, every employee can upload encrypted data on the cloud storage server without the knowledge of the company's master-secret key. Therefore, the best solution for the above problem is that Alice encrypts files with separate public-keys, but Alice sends Bob a single constant-size decryption key. The constant size decryption key should be sent via secured channels and kept secret and the small key is always desirable.

2. Existing System

- **Cryptographic Keys for Predefined Hierarchy**

Cryptographic key assignment schemes aim is to minimize the expense in storing and managing secret keys for general cryptographic use. Utilizing a tree structure, a key for a given branch can use to derive the keys of its descendant nodes. More advanced cryptographic key assignment schemes support access policy that can be modelled by an acyclic graph or a cyclic graph.

- **Compact Key Symmetric-Key Encryption**

Benaloh et.al. [2] Presented an encryption scheme which is originally proposed for concisely transmitting a large number of keys in broadcast scenario [3]. The construction is simple and we briefly review its key derivation process here for a concrete description of what are the desirable properties we want to achieve. The derivation of the key from a set of classes (which is a subset of all possible ciphertext classes) is as follows. A composite modulus is chosen where p and q are two large random primes. A master secret key is chosen at random. Each class is associated with a distinct prime. All these prime numbers can be put in the public system parameter. A constant-size key set can be generated. For those who have been delegated the access rights for S' can be generated. However, it is designed for the symmetric-key setting instead. The content provider needs to get the corresponding secret keys to encrypt data, which is not suitable for many applications. Because method is used to generate a secret value rather than a pair of public/secret keys, it is unclear how to apply this idea for public-key encryption scheme. Finally, we note that there are schemes which try to reduce the key size for achieving authentication in symmetric-key encryption, e.g., [4]. However, the sharing of decryption power is not a concern in these schemes.

- **Compact Key in Identity-Based Encryption (IBE)**

Identity-based encryption (e.g., [5], [6], [7]) 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. There is a trusted party called a private key generated in IBE which holds a master-secret key and distribute a secret key to each user with respect to the user identity. The encryptor can take the public parameter and a user identity to encrypt a message. The receiver can decrypt this ciphertext by his secret key. Guo et al. [8], [9] tried to build IBE with key aggregation. Their scheme, 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. 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 (ABE)**

Attribute-based encryption [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.

- **Primitive is Proxy re-encryption (PRE)**

To delegate the decryption power of some ciphertext without sending the secret key to the delegate, a useful primitive is proxy re-encryption (PRE). A PRE scheme allows the sender to delegate to the server the ability to convert the ciphertext encrypted under her public-key into once for the receiver.

2.1 Disadvantages

- Cryptographic keys for a Predefined Hierarchy generally more expensive than "symmetric-key operations" such as pseudorandom function.
- Compact Key in Symmetric-Key Encryption, it is designed for the symmetric-key setting instead. The encryptor needs to get the corresponding secret keys to encrypt data, which is not suitable for many applications. Since their method is used to generate a secret value rather than a pair of public or secret keys, it is unclear how to apply this idea for public key encryption scheme.
- IBE greatly increases the costs of storing and transmitting ciphertexts, which is impractical in many situations such as shared cloud storage.
- Under ABE, 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]).
- Using PRE just moves the secure key storage requirement from the delegate to the proxy. It is, thus, undesirable to let the proxy reside in the storage server.

3. Proposed System

The proposed method is to design an efficient public-key encryption scheme which supports flexible delegation in the sense that any subset of the ciphertexts is decryptable by a constant-size decryption key and to study how to make a decryption key more powerful in the sense that it allows decryption of multiple ciphertexts, without increasing its size. In proposing a methodology we are introducing a special type of public-key encryption, which is called as a key - aggregate cryptosystem (KAC).

The properties of Key aggregate cryptosystems are decryption key size and cipher text size is constant and KAC is public-key encryption technique. In KAC, users encrypt a message not only under a public-key, but also under an identifier of the ciphertext called class. That means the ciphertexts are further categorized into different classes. The owner holds the key is called as master-secret key, which can be used to extract secret keys for different classes. More importantly, the extracted key is an aggregate key which is as compact as a secret key for a single class, but the aggregate key has the decryption power for any subset of ciphertext classes [1]. This aggregate key can be sent to the receiver via a secured channel like e-mail. Then receiver can download the encrypted content and then use this aggregate key to decrypt these encrypted content or data

3.1 Sharing Encrypted Data

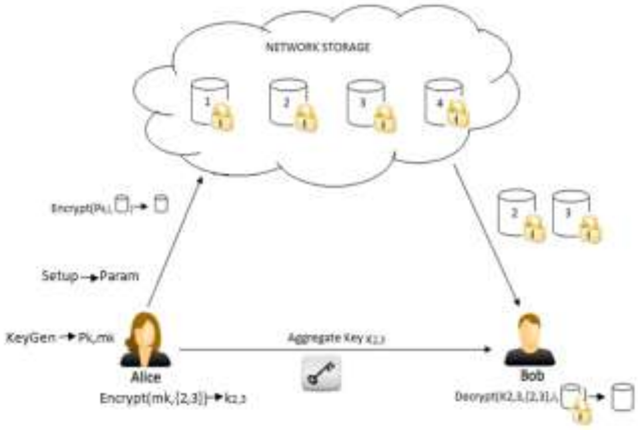

Figure 1: Data sharing in the cloud using key aggregate cryptosystem.

4. Modules

- Setup phase
- Encryption phase
- Key generation phase
- Decryption phase

4.1 Description of Modules

- Setup phase: The setup algorithm takes no input other than the implicit security parameter. It outputs the public parameters PK and a master key MK.
- Key Generation: Key generation is an important part where we have to generate both public key and private key. The sender will be encrypting the message with the message's public key and the receiver will decrypt its secret key.

Now, we have to select a number'd' within the range of 'n'. Using the following equation, we can generate the public key.

Q = d * P

d = the random number that we have selected within the range of (1 to n-1). P is the point on the curve. 'Q' is the public key and'd' is the private key.

- Encryption: Let 'm' be the message that we are sending. We have to represent this message on the curve. Consider *'m'* has the point *'M'* on the curve *'E'*. Randomly select 'k' from [1 to (n-1)].
- Two cipher texts will be generated let it be C1 and C2.
  C1 = k*P
  C2 = M + k*Q
- Decryption: We have to get back the message 'm' that was sent to us,
  M = C2 – d * C1

M is the original message that we have sent.

4.2 Advantages

- A decryption key is more powerful in the sense that it allows decryption of multiple ciphertexts, without raising its size.
- The size of the master-secret key, ciphertext, public-key, and aggregate key in our KAC schemes all are kept constant size.
- A canonical application of KAC is an efficient data sharing scheme.
- The key aggregate property is especially useful the delegation key to be efficient and flexible.
- It is easy to key management.
- Only the particular member can view their messages.
- High security for data sharing.

Table 1: Comparison between KAC scheme and other related scheme.

Different Schemes	*Ciphertext size*	*Decryption key size*	*Encryption type*
Key assignment schemes	Constant	Non-constant	Symmetric or public-key
Symmetric-key Encryption with compost Key	Constant	Constant	Symmetric key
IBE with compact key	Non-constant	Constant	Public key
Attribute based encryption	Constant	Non-constant	Public key
KAC	Constant	Constant	Public key

5. Conclusion

To share the data securely and flexibly is essential thing in cloud computing. Our approach is more secure and flexible in cloud storage. User prefers to upload their data on cloud and among different users. Outsourcing of data to server may lead to leak the private data of users to everyone. For this problem encryption is a one of the best solution which provides to share selected data with desired candidate. Sharing of decryption keys in secure way plays an important role. User data privacy is a central question of cloud storage. Compress secret keys in public-key cryptosystem which support delegation of secret keys for different ciphertext in cloud storage. No matter which one among the power set of classes, the delegate can always get an aggregate key of constant size.

6. Future Work

The future work of our project when one carries the delegated keys around in a mobile device without using special trusted hardware, the key is prompt to leakage, designing a leakage resilient cryptosystem yet allows efficient and flexible key delegation is also interesting direction. For future extension it is necessary to reserve enough ciphertexts class because in cloud ciphertext grows rapidly and the limitation is that predefined bound of the number of maximum ciphertext classes. We introduce an SSH key, Digital signature algorithm for secure authentication in the cloud

7. Acknowledgements

I would like to thank Asst. Prof. Gopal B. I am so grateful for his help, professionalism, and valuable guidance throughout this paper. This accomplishment would not have been possible without his support. Thank you.

References

[1] Cheng-Kang Chu, Chow, S.S.M, 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. *Volume: 25, Issue: 2.* Year: 2014.

[2] 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.

[3] J. Benaloh, — *“Key Compression and Its Application to Digital Fingerprinting,”* Microsoft Research, Tech. Rep., 2009.

[4] B. Alomair and R. Poovendran, — *“Information Theoretically Secure Encryption with Almost Free Authentication,”* J. UCS, *vol. 15, no. 15*, pp. 2937–2956, 2009.

[5] D. Boneh and M. K. Franklin, — *“Identity-Based Encryption from the Weil Pairing,”* in Proceedings of Advances in Cryptology – CRYPTO ’01, ser. LNCS, *vol. 2139*. Springer, 2001, pp. 213–229.

[6] A. Sahai and B. Waters, — *“Fuzzy Identity-Based Encryption,”* in Proceedings of Advances in Cryptology - EUROCRYPT ’05, ser. LNCS, *vol. 3494*. Springer, 2005, pp. 457–473.

[7] S. S. M. Chow, Y. Dodis, Y. Rouselakis, and B. Waters, — *“Practical Leakage-Resilient Identity-Based Encryption from Simple Assumptions,”* in ACM Conference on Computer and Communications Security, 2010, pp. 152–161.

[8] F. Guo, Y. Mu, and Z. Chen, — *“Identity-Based Encryption: How to Decrypt Multiple Ciphertexts Using a Single Decryption Key,”* in Proceedings of Pairing-Based Cryptography (Pairing ’07), ser. LNCS, *vol. 4575*. Springer, 2007, pp. 392–406.

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

[10] S. S. M. Chow, Y. Dodis, Y. Rouselakis, and B. Waters, — *“Practical Leakage-Resilient Identity-Based Encryption from Simple Assumptions,”* in ACM Conference on Computer and Communications Security, 2010, pp. 152–161.

[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.

[12] 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.

[13] T. Okamoto and K. Takashima, — *“Achieving Short Ciphertexts or Short Secret-Keys for Adaptively Secure General Inner-Product Encryption,”* in Cryptology and Network Security (CANS ’11), 2011, pp. 138–159.

[14] 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. 362375, 2013.

Author Profile

Sakkubai M Horatti is Trainee. She has done B.E, M.Tech (Computer Science and Engineering) and having Experience: 4 months + (Trainee in Company) Sakkubai M Horatti has worked in the area of cloud networking which includes the finding constant-size aggregate key for efficient and secured data sharing in cloud storage. And building the different application in cloud using the particular programming language and designing the application done the best work in industry.

Mr. Gopal B is Assistant Professor-I. He has doneB.E, M.Tech (Information Security) and have experience of 1 Year