ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

Cryptographic Algorithms for Efficient and Secure Data Sharing in Cloud Storage

Sweta Kori
PG Scholar,
Computer Science and Engineering,
Bheema institute of Technology and Science.

G.S.Uday Kiran Babu
Assistant Professor,
Computer Science and Engineering,
Bheema institute of Technology and Science.

ABSTRACT:

Data sharing is an important functionality in cloud storage. In this paper, we show how to securely, efficiently, and flexibly share data with others in cloud storage. We describe new public-key cryptosystems that produce constant-size ciphertexts such that efficient delegation of decryption rights for any set of ciphertexts are possible. The novelty is that one can aggregate any set of secret keys and make them as compact as a single key, but encompassing the power of all the keys being aggregated. In other words, the secret key holder can release a constant-size aggregate key for flexible choices of ciphertext set in cloud storage, but the other encrypted files outside the set remain confidential. This compact aggregate key can be conveniently sent to others or be stored in a smart card with very limited secure storage. We provide formal security analysis of our schemes in the standard model. We also describe other application of our schemes. In particular, our schemes give the first public-key patient-controlled encryption for flexible hierarchy, which was yet to be known.

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 contrast, in asymmetric key encryption different keys are used, public key for encryption and private key for decryption. Using asymmetric key encryption is more flexible for our approach. This can be illustrated by following example. Suppose Alice put all data on Box.com and she does not want to expose her data to everyone. Due to data leakage possibilities she does not trust on privacy mechanism provided by Box.com, so she encrypt all data before uploading to the server. If Bob ask her to share some data then Alice use share function of Box.com. But problem now is that how to share encrypted data. There are two severe ways: 1. Alice encrypt data with single secret key and share that secret key directly with the Bob.2. Alice can encrypt data with distinct keys and send Bob corresponding keys to Bob via secure channel. In first approach, unwanted data also get expose to the Bob, which is inadequate.

In second approach, no. of keys is as many as no. of shared files, which may be hundred or thousand as well as transferring these keys require secure channel and storage space which can be expensive. Therefore best solution to above problem is Alice encrypts data with distinct public keys, but send single decryption key of constant size to Bob. Since the decryption key should be sent via secure channel and kept secret small size is always enviable. To design an efficient public key encryption scheme which supports flexible delegation in the sense that any subset of the cipher texts Obviously, the first method is inadequate since all unchosen data may be also leaked to Bob.

For the second method, there are practical concerns on efficiency. The number of such keys is as many as the number of the shared photos, say, a thousand. Transferring 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. Encryption keys also come with two flavors symmetric key or asymmetric (public) key. Using symmetric encryption, when Alice wants the data to be originated from a third party, she has to give the encryptor hersecret key; obviously, this is not always desirable.

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 enterprise settings, 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 distinct public-keys, but only sends Bob a single (constant-size) decryption key. Since the decryption key should be sent via a secure channel and kept secret, small key size is always desirable. For example, we can not expect large storage for decryption keys in the resource-constraint devices like smart phones, smart cards or wireless sensor nodes.Especially, these secret keys are usually stored in the tamper-proof memory, which is relatively expensive. The present research efforts mainly focus on minimizing the communication requirements (such as bandwidth, rounds of communication) like aggregate signature [6]. However, not much has been done about the key itself (see Section 3 for more details).

RELATED WORK:

SYMMETRIC-KEY ENCRYPTION WITH COMPACT KEY:

Benaloh et al. [2] presented an encryption scheme which is originally proposed for concisely transmitting large number of keys inbroadcastscenario [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 for a set of classes (which is a subset of all possible ciphertext cl asses) isas 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 primenumbers can be put in the public system parameterA constantsize key for set can be generated. For those who have been delegated the access rights forScan be generated However, it is designed for the symmetrickey setting instead.

The content provide 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 insymmetric key encryption, e.g., []. However, sharingof decryption power is not a concern in these schemes.

IBE WITH COMPACT KEY:

Identity based encryption (IBE) (e.g., 6], [7]) is apublic key encryption in which the public-key ofa 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 keysto be aggregated must come from different—identitydivisions While there are an exponential number ofidentities and thus secret keys, only a polynomial numberof them can be aggregated.

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

.[1]This significantlyincreases the costs of storing and transmitting ciphertexts, which is impracticalin many situations such as shared cloud storage. AsAnother way to do this is to apply hash function to the string denoting the class, and keep has hing repeatedly until a prime isobtained 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 foran arbitrary set of identities and therefore it does notmatch with our idea of key aggregation.

ATTRIBUTEBASED ENCRYPTION:

Attributebased encryption (ABE)11], [12] allows eachciphertext to be associated with an attribute, and themastersecret key holder can extract a secret key for apolicy of these attributes so that a ciphertext can bedecrypted by this key if its associated attribute conforms policy. For example, with the secret key for thepolicy (13 6 8), one can decrypt ciphertext taggedwith class 1,3,6 or 8. However, the major concern in ABE is

collusion:

resistance but not the compactness ofsecret keys. Indeed, the size of the key often increaseslinearly with the number of attributes it encompasses,or the ciphertextsize is not constant (e.g., [13])

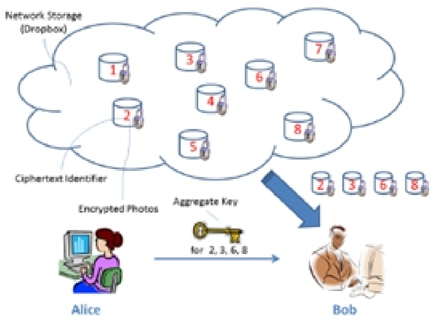

Fig. 1. Alice shares files with identifiers 2, 3, 6 and 8 with Bob by sending him a single aggregate key.

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 notonlyunder a public-key, but also under an identifier of ciphertext called class.

Thatmeanstheciphertexts 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 aggregatekey 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 ciphertextclasses. With our solution, Alice can simply send Bob a singleaggregate key via a secure e-mail. Bob can downloadthe encrypted photos from Alice's Dropbox space andthen use this aggregate key to decrypt these encryptedphotos. The scenario is depicted in Figure 1.The sizes of ciphertext, public-key, master-secret keyand aggregate key in our KAC schemes are all of constant size. The public system parameter has size linear in thenumber of ciphertext classes, but only a small part of it is needed each time and it can be fetched on demand from large (but non-confidential) cloud storage. Previous results may achieve a similar property featuring a constant-size decryption key, but the classes need to conform to some pre-defined hierarchical relationship.

Our work is flexible in the sense that this constraint is eliminated, that is, no special relation is requiredbetween the classes. The detail and other related works can be found in Section 3We propose several concrete KAC schemes with different security levels and extensions in this article. All 3 constructions can be proven secure in the standard model. To the best of our knowledge, our aggregation mechanism2 in KAC has not been investigated.

KEYAGGREGATE CRYPTOSYST EM:

In key aggregate cryptosystem (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. [1] With our example, Alice can send Bob a single aggregate key through a secure email. Bob can download the encrypted photos from Alice's Box.com space and then use this aggregate key to decrypt these encrypted data.

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

The sizes of ciphertext, public key, master secret key and aggregate key in KAC schemes are all of constant size. The public system parameter has size linear in the number of ciphertext classes, but only a small part of it is needed each time and it can be fetched on demand from large (but non confidential) cloud storage.

FRAMEWORk:

The data owner establishes the public system parameter hrough Setup and generates a public/master secret key pair through KeyGen Data mcan be encrypted via Encryptby anyone who also decides what ciphertext class is associated with the plaintext message to be encrypted. The data owner can use the master secret key pa to generate an aggregate decryption key for a set of ciphertext, classes through Extract.The generated keys can be passed to delegatees securely through s ecure emails or secure devices Finally, any user with an aggregate key can decryptany 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:1Setup(1λ n) : The data owner establish public systemparameterviaSetup.On input of a security level parameter 1λ number of iphertext classes n ,itoutputs the public system parameter param 2KeyGen: It is executed by data owner to randomly generate a public/ master-secret key pair (P k msk).3Encrypt(pk, i, m) :It is executed by data owner and for message m and index i ,it computes the ciphertext as C.4Extract (msk, S): It is executed by dat a 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 whoreceived, an aggregate key Ks generated by Extract. On input Ks, set S, an index i denoting theciphertext class ciphertext C belongs to and output is decrypted result m.

3.3 Compact Key in Identity-Based Encryption:

Identity-based encryption (IBE) (e.g., [20], [21], [22]) is a type of public-key encryption in which the public-key of a user can be set as an identity-string of the user (e.g.,an email address). There is a trusted party called private key generator (PKG) in IBE which holds a master-secret key and issues a secret key to each user with respectto the user identity.

The encryptor can take the public parameter and a user identity to encrypt a message. The recipient can decrypt this ciphertext by his secret key.Guo et al. [23], [9] tried to build IBE with key aggregation. One of their schemes [23] assumes random oracles but another [9] does not. In their schemes, key aggregation is constrained in the sense that all keysto 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. Most importantly, theirkey-aggregation [23], [9] comes at the expense of O(n) sizes for both ciphertexts and the public parameter, where n is the number of secret keys which can be aggregated into a constant size one.

This greatly increases the costs of storing and transmitting ciphertexts, which is impractical in many situations such as shared cloud storage. Aswe mentioned, our schemes feature constant ciphertext size, and their security holds in the standard model. In fuzzy IBE [21], 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.

3.4 Other Encryption Schemes:

Attribute-based encryption (ABE) [10], [24] 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 (2 _ 3 _ 6 _ 8), one can decrypt ciphertext taggedwith class 2; 3; 6 or 8. However, the major concern in ABE is collusion-resistance but not the compactness of secret keys. Indeed, the size of the key often increases linearly with the number of attributes it encompasses, or the ciphertext-size is not constant (e.g., [25]). To delegate the decryption power of some ciphertexts without sending the secret key to the delegatee, a useful primitive is proxy re-encryption (PRE) (e.g., [26], [27], [28], [29]). A PRE scheme allows Alice to delegate to the server (proxy) the ability to convert the ciphertexts encrypted under her public-key into ones for Bob. PRE is well known to have numerous applications including cryptographic file system [30].

Nevertheless, Alice has to trust the proxy that it only converts ciphertexts according to her instruction, which is what we want to avoid at the first place. Even worse, if the proxy colludes with Bob, some form of Alice's secret key can be recovered which can decrypt Alice's (convertible) ciphertexts without Bob's further help. That also means that the transformation key of proxy should be wellprotected.

Using PRE just moves the secure key storage requirement from the delegatee to the proxy. It is thus undesirable to let the proxy reside in the storage server. That will also be inconvenient since every decryption requires separate interaction with the proxy. 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

CONCLUSION AND FUTURE WOR:

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. In this article, we consider how to "compress" secret keys in public-key cryptosystems which support delegation of secret keys for different ciphertext classes in cloud storage. No matter which one among the power set of classes, the delegatee 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. 10A limitation in our work is the predefined bound of the number of maximum ciphertext classes.

In cloud storage, the number of ciphertexts usually grows rapidly. So we have to reserve enough ciphertext classes for the future extension. Otherwise, we need to expand the public-key as we described in Section 4.2. Although the parameter can be downloaded with ciphertexts, it would be better if its size is independent of the maximum number of ciphertext classes. On the other hand, when one carries the delegated keys around in a mobile device without using special trusted hardware,the key is prompt to leakage, designing a leakageresilient cryptosystem [22], [34] yet allows efficient and flexible key delegation is also an interesting direction

REFERENCES:

[1] 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.

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

[3] 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.

[4] 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.

[5] 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: From Theory to Applications - Essays Dedicated to Jean-Jacques Quisquater on the Occasion of His 65th Birthday, ser. LNCS, vol. 6805. Springer, 2012, pp. 442–464.

[6] 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.

[7] 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.

[8] 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.

[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.n 4990.n Springer, 2007, pp. 384–398.

[10] 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 CommunicationsSecurity (CCS '06). ACM, 2006, pp. 89–98.