SCALABLE DATA SHARING IN CLOUD STORAGE USING KEY-AGGREGATE CRYPTOSYSTEM

Manasa K.R[1], Babitha M.N[2]

[1]*MTech, CSE, Sri Siddhartha Institute of Technology, Tumkur (India)*

[2]*Asst. prof, Dept of CSE, Sri Siddhartha Institute of Technology, Tumkur (India)*

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 cipher texts such that efficient delegation of decryption rights for any set of cipher texts is 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 cipher text 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.

Keywords: Cloud Storage, Data Sharing, Key-Aggregate Encryption, Patient-Controlled Encryption

I. INTRODUCTION

CLOUD storage is gaining popularity recently. In enterprise settings, there is a rise in demand for data outsourcing, which assists in the strategic management of corporate data. It is also used as a core technology behind many online services for personal applications. Nowadays, it is easy to apply for free accounts for email, photo album, file sharing and/or remote access, with storage size more than 25 GB (or a few dollars for more than 1 TB). Together with the current wireless technology, users can access almost all of their files and emails by a mobile phone in any corner of the world.

By using Key Aggregate Cryptosystem, we can generate unique aggregate keys for a single file or multiple files. In existing system, they generate unique keys for a single file, but in this paper, we propose aggregate key for multiple files. Data sharing is an important functionality in cloud storage. For example, bloggers can let their friends view a subset of their private pictures, an enterprise may grant her employees access to a portion of sensitive data. The challenging problem is how to effectively and efficiently share encrypted data. Of course users can download the encrypted data 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. However, finding an efficient and secure way to share partial data in cloud storage 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 data leakage possibility Alice cannot feel relieved by just relying on the privacy protection mechanisms provided by Dropbox, so she encrypts all the photos using her own keys before uploading. 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 function of Dropbox, but the problem now is how to delegate the decryption rights for these photos to Bob. A possible option Alice can choose is to securely send Bob the secret keys involved. Naturally, there are two extreme ways for her under the traditional encryption paradigm. Alice encrypts all files with a single encryption key and gives Bob the corresponding secret key directly. Alice encrypts files with distinct keys and sends Bob the corresponding secret keys. 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 encrypter her secret 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 cannot 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 [5]. However, not much has been done about the key itself.

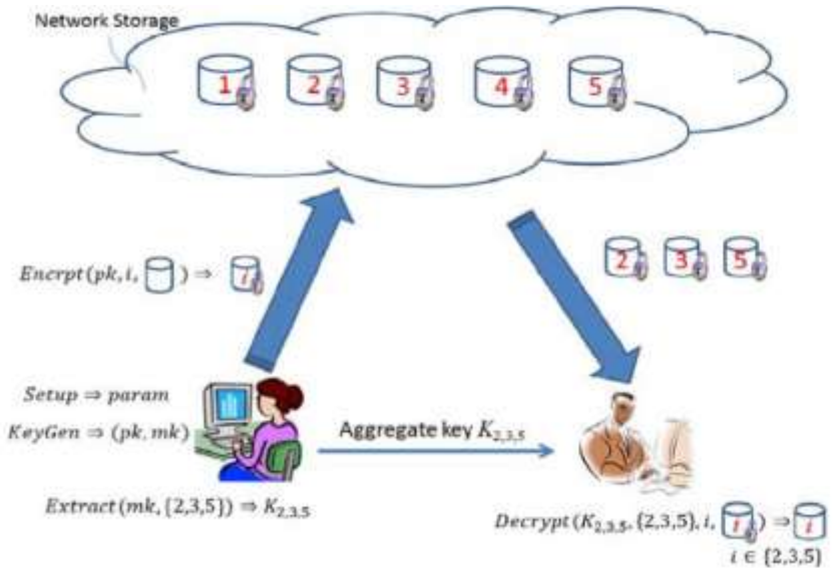
Network Storage
1
2
3
4
5
$Encrpt(pk, i, \square) \Rightarrow i$
2
3
5
$Setup \Rightarrow param$
$KeyGen \Rightarrow (pk, mk)$
Aggregate key $K_{2,3,5}$
$Extract(mk, \{2,3,5\}) \Rightarrow K_{2,3,5}$
$Decrypt(K_{2,3,5}, \{2,3,5\}, i, i) \Rightarrow i$
$i \in \{2,3,5\}$

Fig 1: Using KAC for Data Sharing in Cloud Storage

II. KEY-AGGREGATE ENCRYPTION

In this paper, we study how to make a decryption key more powerful in the sense that it allows decryption of multiple cipher texts, without increasing its size. Specifically, our problem statement is:
“To design an efficient public-key encryption scheme, this supports flexible delegation in the sense that any subset of the cipher texts (produced by the encryption scheme) is decryptable by a constant-size decryption key (generated by the owner of the master-secret key).”
We solve this problem by introducing a special type of public-key encryption which we call key-aggregate crypto system (KAC). In KAC, users encrypt a message not only under a public-key, but also under an identifier of cipher text called class. That means the cipher texts 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 cipher text classes. With our solution, Alice can simply send Bob a single aggregate key via a secure e-mail. Bob can download the encrypted photos from Alice’s Dropbox space and then use this aggregate key to decrypt these encrypted photos. The scenario is depicted in Fig. 1.
A key-aggregate encryption scheme consists of five polynomial-time algorithms as follows. The data owner establishes the public system parameter via Setup and generates a public/master-secret3 key pair via KeyGen. Messages can be encrypted via Encrypt by anyone who also decides what cipher text class is associated with the plaintext message to be encrypted. The data owner can use the master-secret to generate an aggregate decryption key for a set of cipher text classes via Extract. The generated keys can be passed to delegates securely (via secure e-mails or secure devices) finally, any user with an aggregate key can decrypt any cipher text provided that the cipher text’s class is contained in the aggregate key via Decrypt.

- **Setup (1^λ, n):** Executed by the data owner to setup an account on an untrusted server. On input a security level parameter 1^λ and the number of cipher text classes **n** (i.e., class index should be an integer bounded by 1 and **n**), it outputs the public system parameter param, which is omitted from the input of the other algorithms for brevity.
- **KeyGen**: executed by the data owner to randomly generate a public/master-secret key pair (pk, msk).
- **Encrypt (pk, i, msk):** executed by anyone who wants to encrypt data. On input a public-key **pk**, an index i denoting the cipher text class, and a message **m**, it outputs a cipher text **C**.
- **Extract (msk, S):** executed by the data owner for delegating the decrypting power for a certain set of cipher text classes to a delegatee. On input the master-secret key msk and a set **S** of indices corresponding to different classes, it outputs the aggregate key for set **S** denoted by KS.
- **Decrypt (KS, S, I, C):** executed by a delegatee who received an aggregate key KS generated by **Extract**. On input KS, the set **S**, an index i denoting the cipher text class the cipher text **C** belongs to, and **C**, it outputs the decrypted result m if **I € S.**

III.RELATED WORK

[1] In a proxy re-encryption (PRE) scheme, special information is given to the proxy that allows it to translate a cipher text under one key into a cipher text of the same message under a different key. The proxy did not learn

anything about the messages that was encrypted under either key. PRE schemes have many practical applications, including distributed storage, email, and DRM.

[2] This paper proposed a new variant of PRE, named TR-CPBRE, which achieves conditional delegation, broadcast re-encryption and timed-release property simultaneously. In this paper for the first time they introduced a new notion Timed Release Conditional Proxy Broadcast Re-Encryption (TR-CPBRE).We also showed that our scheme can be proved IND-sID-CCA secure in the random oracle model under the (P; Q; f)-GDDHE assumption. This paper also motivates some interesting open problems, for example, how to construct a CCA-secure TR-CPBRE scheme in the adaptive identity model, i.e. achieving IND-aID-CCA security.

[3] Users should be able to use the cloud storage without worrying about the need to verify its integrity. Thus, enabling public auditability for cloud storage is of critical importance so that users can resort to a third-party auditor (TPA) to check the integrity of outsourced data and be worry free. To securely introduce an effective TPA, the auditing process need not bring new vulnerabilities towards user data privacy, and need not introduce additional online burden to user. This paper proposed a privacy-preserving public auditing system for data storage security in cloud computing. They utilize the homomorphic linear authenticator and random masking to guarantee that the TPA would not learn any knowledge about the data content stored on the cloud server during the efficient auditing process, which not only eliminates the burden of cloud user from the tedious and possibly expensive auditing task, but also alleviates the users' fear of their outsourced data leakage.

[4] This paper, introduced a right approach to achieve anonymity in storing data to the cloud with publicly verifiable data-integrity in mind. This approach decouples the anonymous protection mechanism from the provable data possession mechanism via the use of security mediator (SEM). This solution not only minimizes the computation and bandwidth requirement of this mediator, but also minimizes the trust placed on it in terms of data privacy and identity privacy. The efficiency of the system is also empirically demonstrated. The distinctive features of this scheme also include data privacy, such that the SEM does not learn anything about the data to be uploaded to the cloud at all, and thus the trust on the SEM is minimized. In addition, this scheme extends their work with the multi-SEM model, which can avoid the potential single point of failure. Security analyses prove that this scheme is secure and efficient.

[5] An aggregate signature scheme is a digital signature that supports aggregation. Given n signatures on n distinct messages from n distinct users, it is possible to aggregate all these signatures into a single short signature. This single signature (and the n original messages) will convince the verifier that the n users did indeed sign the n original messages. This paper introduced the concept of an aggregate signature, present security models for such signatures, and gives several applications for aggregate signatures. They construct an efficient aggregate signature from a recent short signature scheme based on bilinear maps. Aggregate signatures are useful for reducing the size of certificate chains and for reducing message size in secure routing protocols such as SBGP. This paper also shows that aggregate signatures give rise to verifiably encrypted signatures. Such signatures enable the verifier to test that a given cipher text C is the encryption of a signature on a given message M. Verifiably encrypted signatures are used in contract-signing protocols. Finally, this paper shows that similar ideas can be used to extend the short signature scheme to give simple ring signatures.

IV. COMPARISION

In PRE scheme, previously proposed re-encryption schemes achieved only semantic security in contrast, applications often require security against chosen cipher text attacks. They propose a definition of security against chosen cipher text attacks for PRE schemes, and present a scheme that satisfies the definition. Their construction is efficient and based only on the Decisional Bilinear Diffe-Hellman assumption in the standard model. It also formally captures CCA security for PRE schemes via both a game-based definition and simulation-based definitions that guarantee universally composable security.

In a Conditional Proxy Broadcast Re-encryption scheme, when compared with the existing CPBRE and TR-PRE schemes, this scheme not only requires less number of pairings and achieves better efficiency in communication, but also enables the delegator to make a fine-grained delegation of decryption rights to multiple delegatees without losing CCA security.

In privacy preserving public auditing scheme compared to previous one, considering TPA may concurrently handle multiple audit sessions from different users for their outsourced data files, this scheme further extend their privacy-preserving public auditing protocol into a multiuser setting, where the TPA can perform multiple auditing tasks in a batch manner for better efficiency. Extensive analysis shows that these schemes are provably secure and highly efficient.

In SEM scheme, previous methods either unnecessarily reveal the identity of a data owner to the untrusted cloud or any public verifiers, or introduce significant overheads on verification metadata for preserving anonymity. This paper proposed a simple, efficient, and publicly verifiable approach to ensure cloud data integrity without sacrificing the anonymity of data owners not requiring significant overhead. Specifically, we introduce a security-mediator (SEM), which is able to generate verification metadata (i.e., signatures) on outsourced data for data owners.

In Aggregate signature scheme, previous signature constructions using bilinear maps only required a gap Diffie-Hellman group (i.e., DDH easy, but CDH hard). The signature constructions in this paper require the extra structure provided by the bilinear map. These constructions are an example where a bilinear map provides more power than a generic gap Diffie-Hellman group.

V. CONCLUSION

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 paper, we consider how to "compress" secret keys in public-key cryptosystems which support delegation of secret keys for different cipher text 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. Key Aggregate cryptosystem is an efficient method and it also reduces cost.

REFERENCES

[1] G. Ateniese, K. Fu, M. Green, and S. Hohenberger, "Improved Proxy Re-Encryption Schemes with Applications to Secure Distributed Storage," ACM Trans. Information and System Security, vol. 9, no. 1, pp. 1-30, 2006.

[2] C.-K. Chu, J. Weng, S.S.M. Chow, J. Zhou, and R.H. Deng, "Conditional Proxy Broadcast Re-Encryption" Proc. 14th Australasian Conf. Information Security and Privacy (ACISP '09), vol. 5594, pp. 327-342, 2009.

[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, Feb. 2013.

[4] B. Wang, S.S.M. Chow, M. Li, and H. Li, "Storing Shared Data on the Cloud via Security-Mediator," Proc. IEEE 33rd Int'l Conf. Distributed Computing Systems (ICDCS), 2013.

[5] D. Boneh, C. Gentry, B. Lynn, and H. Shacham, "Aggregate and Verifiably Encrypted Signatures from Bilinear Maps," Proc. 22nd Int'l Conf. Theory and Applications of Cryptographic Techniques. (EUROCRYPT '03), pp. 416-432, 2003.