Efficient Data Sharing in Cloud Medium with Key Aggregate Cryptosystem

C. Suganya[1], A. Sangeetha[2]

[1, 2] Mount Zion College of Engineering and Technology, Pudukottai, Anna University, India

Abstract: ***In Cloud Storage there is an important functionality called Data Sharing, but the query always present in every one's mind is how to securely, efficiently, and flexibly share data with others in cloud storage. A new public-key cryptosystem is introduced to produce a constant size cipher texts such that efficient allocation of decryption rights for any set of cipher texts are possible. The uniqueness is that one can aggregate any set of secret keys and make them as compact as a single key, but surrounding the power of all the keys being aggregated. In supplementary terminology, the secret key owner 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 private. This packed together aggregate key can be suitably sent to others or be stored in owner's end to make the process more secure. Once the key is used by the owner then the formal key system change the key aggregate strategies and generates a new key for the data decryption, so that the user can access the remote resource with the help of generated key aggregate at single time only after that a new key will be generated for further use. In particular, this approach gives the first public key patient controlled encryption for flexible hierarchy, which was yet to be known.***

Keywords: Cloud Storage, Data Sharing, Public-key Cryptosystem, Secret key, Aggregate key

1. Introduction

Cloud storage is gaining popularity recently. In enterprise settings, we see the 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, and file sharing with storage size more than 25GB (or a few dollars for more than 1TB). 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. Considering data privacy, a traditional way to ensure it is to rely on the server to enforce the access control after authentication, which means any unexpected privilege escalation will expose all data.

In a shared-tenancy cloud computing environment, things become even worse. Data from different clients can be hosted on separate virtual machines (VMs) but reside on a single physical machine. Data in a target VM could be stolen by instantiating another VM co-resident with the target one. Regarding availability of files, there are a series of cryptographic schemes which go as far as allowing a third-party auditor to check the availability of files on behalf of the data owner without leaking anything about the data, or without compromising the data owner's anonymity. Likewise, cloud users probably will not hold the strong belief that the cloud server is doing a good job in terms of confidentiality. A cryptographic solution, with proven security relied on number-theoretic assumptions is more desirable, whenever the user is not perfectly happy with trusting the security of the VM or the honesty of the technical staff. These users are motivated to encrypt their data with their own keys before uploading them 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 pictures; an enterprise may grant her employees access to a portion of sensitive data. The challenging problem is how to effectively 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. Assume that Alice puts all her private photos on Drop box, 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 Drop box, 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 Drop box, 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 unclose 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 flavours - 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 encrypt the 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.

2. Related Work

Besides, in order to improve feasibility and save on the expense in the security paradigm, it is preferred to get the information retrieval result with the most relevant keys that match users interest instead of all the keys, which indicates that the keys should be ranked in the order of relevance by users interest and only the keys with the highest relevance are selected by the users. A series of searchable symmetric encryption schemes have been proposed to enable search on cipher text. Traditional schemes enable users to securely retrieve the cipher text, but these schemes support only Boolean keyword search, i.e., whether a key exists in a system or not, without considering the difference of relevance with the queried keys of these encrypted data in the result. Preventing the security from involving in ranking and entrusting all the work to the user is a natural way to avoid information leakage. However, the limited computational power on the user side and the high computational overhead against information security.

A. To improve security without sacrificing efficiency, schemes presented in show that they support top-k single key retrieval under various scenarios.
B. Authors made attempts to solve the problem of top-k multi-keys over encrypted data. These schemes, however, suffer from two problems – Boolean representation and how to strike a balance between security and efficiency.
C. In the former, data are ranked only by the number of retrieved keys, which impairs search accuracy. In the latter, security is implicitly compromised to tradeoff for efficiency, which is particularly undesirable in security-oriented applications.

2.1 Spice - Simple Privacy-Preserving Identity-Management for Cloud Environment

In this paper [8], SPICE – the first digital identity management system that can satisfy these properties in addition to other desirable properties. The novelty of our scheme stems from combining and exploiting two group signatures so that we can randomize the signature to make the same signature look different for multiple uses of it and hide some parts of the messages which are not the concerns of the CSP. Its scheme is quite applicable to cloud systems due to its simplicity and efficiency.

2.2 Secure Computers are not so Secure

According to this paper [5], the requirements for achieving privacy and security in the Cloud and also briefly outlines the requirements for secure data sharing in the Cloud. It provided a survey on privacy and security in the Cloud focusing on how privacy laws should also take into consideration Cloud computing and what work can be done to prevent privacy and security breaches of one's personal data in the Cloud. This explored factors that affect managing information security in Cloud computing. It explains the necessary security needs for enterprises to understand the dynamics of information security in the Cloud.

2.3 Privacy-Preserving Public Auditing for Secure Cloud Storage

This paper focuses on [2], a privacy-preserving public auditing system for data storage security in Cloud Computing. We 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. Considering TPA may concurrently handle multiple audit sessions from different users for their outsourced data files, and its extend our privacy-preserving public auditing protocol into a multi-user setting, where the TPA can perform multiple auditing tasks in a batch manner for better efficiency. Extensive analysis shows that our schemes are provably secure and highly efficient.

2.4 Storing Shared Data on The Cloud via Security

In this paper [1], an organization can employ its own anonymous authentication mechanism, and the cloud is oblivious to that since it only deals with typical PDP-metadata. Consequently, there is no extra storage overhead when compared with existing non-anonymous PDP solutions. The distinctive features of our scheme also include data privacy, such that the SEM does not learn anything about the data to be uploaded to the cloud at all, which is able to minimize the requirement of trust on the SEM. Additionaly, to work with the multi-SEM model, which can avoid the potential single point of failure existing in the single-SEM scenario. Security analyses prove that the scheme is secure, and experiment results demonstrate which scheme is efficient.

3. Proposed Approach

In the Proposed Approach, by utilizing public key based homomorphic authenticator with random masking privacy preserving public auditing can be achieved. The technique of bilinear aggregate signature is used to achieve key auditing. Key auditing reduces the computation overhead. Extensive security and performance analysis shows the proposed schemes are provably secure and highly efficient. Aggregate Key auditing where multiple delegated auditing asks for different keys from different users can be performed simultaneously by the user and also supports dynamic operations on data blocks i.e. data update, append and delete. We introduce the concepts of similarity relevance and scheme robustness to formulate the privacy issues in encryption schemes, and then solve the insecurity problem by proposing a random key encryption scheme. Novel technologies in the cryptography community and information retrieval community are employed, including homomorphic encryption and vector space model. In the proposed scheme, the majority of computing work is done on the encrypted data while the user takes part in ranking, which guarantees top k multi-keys provides efficient retrieval of data over encrypted data with high security and practical efficiency.

A. This scheme fulfills the secure multi-keyword top-k retrieval over encrypted data. Specifically, for the first time we employ relevance score to support multi-keyword top-k retrieval.
B. Thorough analysis on security demonstrates the proposed scheme guarantees high data privacy. Furthermore, performance analysis and experimental results show that our scheme is efficient for practical utilization.

3.1 Query Generation Algorithm

Query Processing is a key determinant in the overall performance of distributed databases. It requires processing of data at their respective sites and transmission of the same between them. These together constitute a distributed query processing strategy (DQP). DQP aims to arrive at an efficient query processing strategy for a given query. This strategy involves generation of efficient query plans for a distributed query. In case of distributed relational queries, the number of possible query plans grows exponentially with an increase in the number of relations accessed by the query.

3.2 Polynomial Time Algorithm

In computer science, the time complexity of an algorithm quantifies the amount of time taken by an algorithm to run as a function of the size of the input to the problem. When expressed this way, the time complexity is said to be described asymptotically, i.e., as the input size goes to infinity. Time complexity is commonly estimated by counting the number of elementary operations performed by the algorithm, where an elementary operation takes a fixed amount of time to perform.

3.3 Encryption/Decryption Algorithm

Regular cryptosystems are ones where mathematical operations on the cipher text have regular effects on the plaintext. A normal symmetric cipher DES, AES, or whatever is not an efficient cipher process. Assume have a plaintext P and encrypt it with AES to get a corresponding cipher text C. If multiply that cipher text by 2, and then decrypt 2C, get random gibberish instead of P. If got something else, like 2P, that would imply some pretty strong non-randomness properties of AES and no one would trust its security. The RSA algorithm is different. Encrypt P to get C, multiply C by 2, and then decrypt 2C -- and get 2P. That's a Superior Cipher Scheme: perform some mathematical operation to the cipher text, and that operation is reflected in the plaintext.

4. System Design

4.1 System Architecture

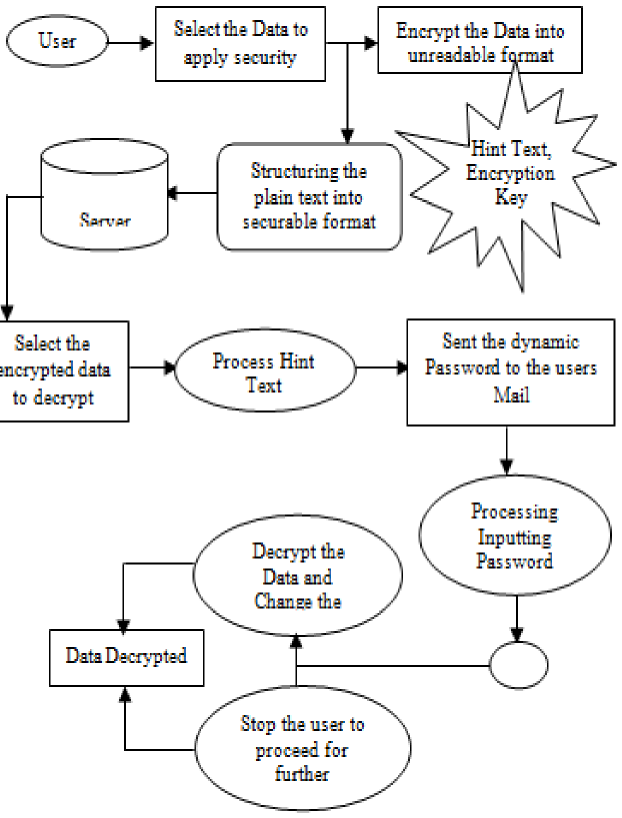
User
Select the Data to apply security
Encrypt the Data into unreadable format
Hint Text, Encryption Key
Server
Structuring the plain text into securable format
Select the encrypted data to decrypt
Process Hint Text
Sent the dynamic Password to the users Mail
Processing Inputting Password
Decrypt the Data and Change the
Data Decrypted
Stop the user to proceed for further

Figure 1: System Architecture

The major part of the project development sector considers and fully survey all the required needs for developing the project. System architecture is the conceptual model that defines the structure, behavior, and more views of a system. An architecture description is a formal description and representation of a system, organized in a way that supports reasoning about the structures and behaviors of the system.

4.2 Modules

1. Key Aggregate Based Encryption

2. Information Retrieval.
3. Hint Text Manipulation.
4. Random Key Analysis.
5. Dynamic Decryption.

4.2.1 Key Aggregate Based Encryption

- Key generation (KG): The algorithm takes as an input a security parameter k and outputs a public and private key pair (pk; sk), where pk is public, while sk is kept secret.
- Encryption (E): The algorithm takes as input a plaintext m 2 f0; 1g and the public key pk, and output a cipher text c, denoted as c ¼ Em;
- Decryption (D): The algorithm takes as input a cipher text c and the private key sk, and outputs a plaintext m 2 f0; 1g, denoted as m ¼ Dc; sk.

4.2.2 Information Retrieval

The generic single database PIR protocol is built on a FHE scheme (KG, E, D, Add, and Mult) and consists of three algorithms (Query Generation QG, Response Generation RG, and Response Retrieval RR). At a high level, the user generates a public and private key pair (pk; sk) for the FHE scheme, sends the public key pk to the database server, but keeps the private key sk secret. Then the user chooses an index i, where 1 < i < n, and encrypts i with the public key pk, and sends the cipher text as a query to the database server. Based on the response generation circuit and Aggregate properties, the server computes an encryption of the ith bit as a response based on the database, the query and the public key pk, and sends the response back. At the end, the user decrypts the response to obtain the i^{th} bit. Assume that the user and the database server have agreed upon a FHE scheme (KG, E, D, Add, and Mult) in advance, our single-database PIR can be using Query generation, Response generation, Response Retrieval.

4.2.3 Hint Text Manipulation

This system fully involves in the context of generating efficient hint texts against the given data. Once the user inputting the data this system, asks the user to provide the hint text for manipulating the data against encryption, after that the hint text and sampling data will be forwarded to the user's mail for clarification.

4.2.4 Random Key Analysis

The Random Key Analysis module takes as an input a security parameter k and outputs a public and private key pair (pk; sk), where pk is public, while sk is kept secret. In order to use a smaller set of cryptographic keys, a sender uses multiple keys to encrypt a message and a receiver needs multiple keys to decrypt the message. Instead of the above mentioned process, this scheme takes a security parameter A and determines a (convenient) parameter set Ap=A, AP=2A, n=(oA2)=r+A, where r is the bit length of the cipher text, n is the bit-length of the secret key, Ap is the bit-length of the noise and oA2 is the number of integers in the public key.

4.2.5 Dynamic Decryption

In cryptography, encryption is the process of transforming information (referred to as plaintext) using an algorithm (called cipher) to make it unreadable to anyone except those possessing special knowledge, usually referred to as a key. The result of the process is encrypted information (in cryptography, referred to as cipher text). In many contexts, the word encryption also implicitly refers to the reverse process, decryption (e.g. "software for encryption" can typically also perform decryption), to make the encrypted information readable again (i.e. to make it unencrypted).

5. Conclusion and Future Work

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. 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 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. A limitation in our work is the predefined bound of the number of maximum cipher text classes. In cloud storage, the number of cipher texts usually grows rapidly. So we have to reserve enough cipher text classes for the future extension. 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. 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 leakage resilient cryptosystem.

6. Acknowledgement

We would like to sincerely thank Assistant Prof. A.Sangeetha for his advice and guidance at the start of this article. His guidance has also been essential during some steps of this article and his quick invaluable insights have always been very helpful. His hard working and passion for

research also has set an example that we would like to follow. We really appreciate his interest and enthusiasm during this article. Finally we thank the Editor in chief, the Associate Editor and anonymous Referees for their comments.

References

[1] 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.
[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] 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.

[4] 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.

[5] L. Hardesty, “Secure computers aren’t so secure,” MIT press,2009,http://www.physorg.com/news176107396. html.

[6] 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.

[7] 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.

[8] 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.

Author Profile

C.Suganya is currently a PG scholar in Software Engineering from the Department of Computer Science at Mount Zion College of Engineering and Technology, Pudukkottai. She received his Bachelor Degree in Computer Science and Engineering from M.I.E.T Engineering College, Tiruchirappalli and Tamilnadu. Her Research areas include Cloud Computing, Grid Computing and Wireless Sensor Networks.

A.Sangeetha is currently working as an Asst. Professor from the Department of Computer Science and Engineering at Mount Zion College of Engineering and Technology, Pudukkottai. She received his Master Degree from Bharathidasan University, Tiruchirappallai and Tamilnadu. Her main research interests lie in the area of Cloud Computing, Distributed Computing and Wireless sensor Networks.