Rungta International Journal of Computer Science and Information Technology
3 (1&2), 2018, 5-8
http://www.publishingindia.com/ijcsit

Multi-Cloud Environment Cryptosystem for Data Sharing

T. V. Pawar[1*], S. R. Badgujar[2], and M. B. Jhade[3]
[1]Lecturer, Dept. of Computer Engineering, K. K. Wagh Polytechnic, Chandori, Maharashtra, India.
Email: tvpawar@kkwagh.edu.in
[2]Lecturer, Dept. of Computer Technology, K. K. Wagh Polytechnic, Nashik, Maharashtra, India.
Email: srbadgujar@kkwagh.edu.in
[3]Principal, K. K. Wagh Women's Polytechnic, Nashik, Maharashtra, India.
Email: mbjhade@kkwagh.edu.in
*Corresponding Author

Abstract: Secure data exchange is an important feature in cloud storage. In this paper, we show how to provide two-stage security, here the cryptographic data is the first phase and the division is another phase. Provides shared data securely, efficiently and flexibly with others in multiple cloud storage using an aggregated cryptosystem. We describe new public-key cryptosystems that produce encrypted texts of constant size, so that an effective delegation of decryption rights is possible for any set of encrypted texts that decrypted files are split and stored in different clouds for security reasons. The novelty is that you can add any set of secret keys and make them compact as a single key, but by understanding the power of all the keys that are added. In other words, the secret key holder can issue an aggregate key of constant size for flexible options of encrypted text sets in the cloud storage, but the other encrypted files outside the set remain confidential. This compact aggregate key can be conveniently sent to others or stored on a smart card with very limited secure storage space. We offer formal security analysis of our schemes in the standard model. We also describe other applications of our schemes. In particular, our schemes provide the first patient-controlled cryptography of the public key for a flexible hierarchy that was not yet known.

Keywords: Authentication, Data accessing, Key gathering, Multi-cloud storage.

I. Introduction

The most important aspect of the cloud is security. Cloud storage is gaining popularity recently. In corporate environments, we see increased demand for data outsourcing, which helps in the strategic management of corporate data. It is also used as a central technology behind many online services for personal applications. Nowadays, it's easy to request free e-mail accounts, photo albums; file exchange and/or remote access, with a storage size exceeding 25 GB. Along with current wireless technology, users can access almost all their files and emails from a mobile phone anywhere in the world. Given the confidentiality of data, a traditional way to ensure that it is relying on servers to enforce access control after authentication, which means that any unexpected scale of privileges will expose all data. In a shared cloud computing environment, things get even worse. Data from different clients can be hosted on separate virtual machines, but reside on a single physical machine. Data in a target virtual machine could be stolen by creating an instance of another VM co-resident with the destination. Regarding file availability, there are a number of cryptographic schemes that allow an external reviewer to check the availability of files on the data owner's account that has not filtered anything on the data, or without compromising the anonymity of the data owners. Likewise, cloud users probably will not have the firm belief that the cloud server is doing a good job in terms of confidentiality. An encryption solution, for example, with proven security based on hypothesis of number theory is more desirable, whenever the user is not fully satisfied with confidence in the virtual machine security 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 [1]. 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. 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 or he has to give 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 [2].

II. Backround (Literature Survey)

As per the survey of the existing systems following points are observed:

- Wen-Guey Tzeng proposed a time-bound cryptographic key assignment scheme in which the cryptographic keys of a class were different for each time period, that was the cryptographic key of class Ci at time r is K(i,t). Key derivation is constrained not only by the class relation, but also the time period. In our scheme, each user holds some secret parameters whose number is independent of the number of the classes in the hierarchy and the total time periods. We present two novel applications of our scheme. One is to broadcast data to authorized users in a multilevel security way and the other is to construct a flexible cryptographic key backup system [3].
- Cong Wang proposed a secure cloud storage system supporting privacy-preserving public auditing. These techniques extend our result to enable the TPA to perform audits for multiple users simultaneously and efficiently. Extensive security and performance analysis show the proposed schemes are provably secure and highly efficient. Our preliminary experiment conducted on Amazon EC2 instance further demonstrates the fast performance of the design [4].
- Xiaoming Hu *et al.* proposed a Gentry's identity-based encryption scheme, we give a construction for an ID-PRE scheme that is fully secure in the standard model. Proposed scheme has the following advantages comparison with all previous ID PRE Schemes: Short Public Parameters, a tight reduction and fully security in standard model [5].
- Laurence T. Yang *et al.* propose a scalable two-phase top-down specialization (TDS) approach to anonymize large-scale data sets using the MapReduce framework on cloud. In both phases of our approach, we deliberately design a group of innovative MapReduce jobs to concretely accomplish the specialization computation in a highly scalable way. Experimental evaluation results demonstrate that with our approach, the scalability and efficiency of TDS can be significantly improved over existing approaches [6].
- Surya Nepal proposed a secure cloud storage service architecture with the focus on Data Integrity as a Service (DIaaS) based on the principles of Service Oriented Architecture and Web services. Our approach not only releases the burdens of data integrity management from a storage service by handling it through an independent third party data Integrity Management Service (IMS), but also reduces the security risk of the data stored in the storage services by checking the data integrity with the help of IMS. We define data integrity protocols for a number of different scenarios, and demonstrate the feasibility of the proposed architecture, service and protocols by implementing them on a public cloud, Amazon S3. We also study the impact of our proposed protocols on the performance of the storage service and show that the benefits of our approach outweigh the little penalty on the storage service performance [7].

III. Proposed System

An aggregate key cryptography scheme consists of five polynomial time algorithms. The data owner sets the public system parameter through the configuration and generates a pair of public / master-secret keys via KeyGen. Anyone who also decides what type of encrypted text is associated with the plaintext message can encrypt messages via Encryption. The data owner can use master secret to generate an aggregated decryption key for a set of text classes that are encrypted using Extract. The generated keys can be passed to the delegates safely (via secure e-mails or secure devices). Finally, any user with an added key can decrypt any encrypted text as long as the encrypted text class is contained in the key added to through Decrypt.

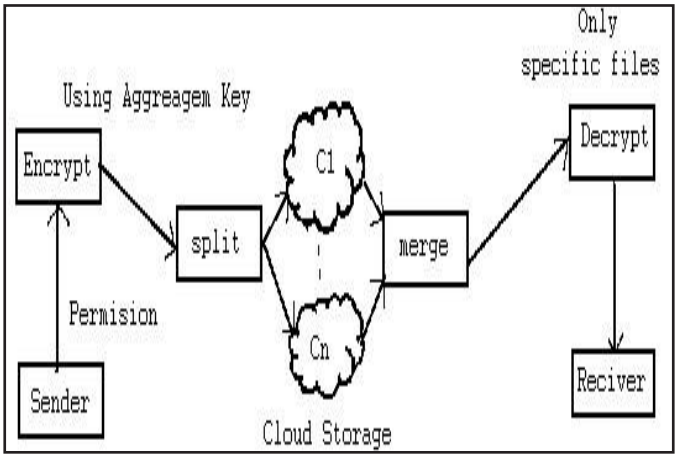

Fig. 1: Architecture of Proposed System

1. *Setup(1, n):* Executed by the data owner to setup an account on an untrusted server. On input a security level parameter 1 and the number of ciphertext classes n (i.e., class index should be an integer bounded by 1 and n), it outputs the public system parameter pram, which is omitted from the input of the other algorithms for brevity.
2. *Permission():* It selects the appropriate files for the specific users. It is one type of access right module.
3. *Add Circle():* It is on type of group. It is used to send a data to the specific users. It saves the time of user to select each user individually. Users have full authority to create its own separate groups or circles to save time and some effort.
4. *KeyGen(pk, msk):* Executed by the data owner to randomly generate a public / master-secret key pair (pk; msk).
5. *Encrypt(pk, i, m):* Executed by anyone who wants to encrypt data. On input a public-key pk, an index we

denoting the cipher text class, and a message m, it outputs a cipher text C.

6. *Merge():* It combines the separated parts of file form the different clouds.
7. *Extract(msk, S):* Executed by the data owner for delegating the decrypting power for a certain set of ciphertext classes to a delegate. 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.
8. *Decrypt(KS, s, i, C):* Executed by a delegate who received an aggregate key KS generated by Extract. On input KS, the set S, an index we denoting the ciphertext class the ciphertext C belongs to, and C, it outputs the decrypted result m if i £ S. User only able to decrypt those files which are accessible to that user.

Algorithm of Proposed System

Step 1. Start.

Step 2. User can register by basic information like username, password, Email-id & Contact number.

Step 3. User can send the request to other users.

Step 4. Only "Request Accepted" users will be appeared in the friend List.

Step 5. Upload a File.

Step 6. Select Users.

Step 7. Give the Permissions.

Step 8. Is selected user is valid?

1. If Yes - Then secure key is sent to that user through mail.

2. If No - Block the user.

Step 9. At Receiver side decrypt the data using receiver key.

Step 10. User will get original file.

Step 11. Stop.

IV. Methodology

A. Cryptographic Keys for a Predefined Hierarchy

Cryptographic key assignment schemes aim 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 be used to derive the keys of its descendant nodes (but not the other way round). Just granting the parent key implicitly grants all the keys of its descendant nodes. We proposed a method to generate a tree hierarchy of symmetric-keys by using repeated evaluations of pseudorandom function / block cipher on a fixed secret. The concept can be generalized from a tree to a graph. More advanced cryptographic key assignment schemes support access policy that can be modeled by an acyclic graph or a cyclic graph. Most of these schemes produce keys for symmetric-key cryptosystems, even though the key derivations may require modular arithmetic as used in public-key cryptosystems, which are generally more expensive than "symmetric-key operations" such as pseudorandom function [8].

B. Compact Key in Identity-Based Encryption (IBE)

IBE 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 in IBE which holds a master-secret key and issues 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 recipient can decrypt this ciphertext by his secret key. We tried to build IBE with key aggregation. One of their schemes assumes random oracles but another does not. 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. Most importantly, their key-aggregation comes at the expense of sizes for both cipher texts 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 cipher texts, which is impractical in many situations such as shared cloud storage. As mentioned, in the schemes feature constant ciphertext size and their security hold in the standard model [9].

In aggregate cryptosystem authentication is necessary for each user in which user login if user login successfully then proceed for further process. User may be sender or receiver. Permission function of sender it gives the permissions like read; write etc. to data for security and proceeds to encryption function. It encrypts data using aggregate key that key size is fixed for every user but it can be generated dynamically. Split function uploads the data but before uploading it splits the encrypted data into different parts and stored that part on different clouds.

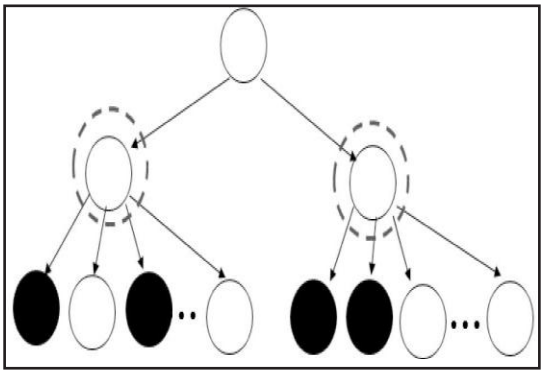

Fig. 2: Key Assignment in our Approach of Proposed System

Here, Merge is the function of receiver side, it retrieves the data from different clouds like C1,C2,C3…Cn. Decrypt function decrypt the date using the private key and aggregate key and proceed for the further processing. Extractor checks if that file is accessible to that user or not. In case valid user then it decrypt from that whole bunch.

V. Conclusion

The secure multi-cloud environment can be used to better protect the privacy of user data in the currently available online services. Using multiple cloud providers to achieve security and privacy benefits is not trivial. The proposed system provides shared data securely, efficiently and flexibly with others in cloud storage. It also stores encrypted data in different clouds to ensure high security. This means that if a cloud is hacked, but some data in steel is protected in another cloud. For this it uses the concept of the n2k algorithm. In every cryptography, every dimension of the key is constant. Our approach is more flexible than assigning hierarchical keys that can only save spaces if all key holders share a similar set of privileges. It also ensures a high level of security by storing files divided into different media clouds if the cloud data is violated, but the steel file is secure.

References

[1] C.-K. Chu, S. S. M. Chow, W.-G. Tzeng, J. Zhou, and R. H. Deng, "Key-aggregate cryptosystem for scalable data sharing in cloud storage," *IEEE Trans. Parallel and Distributed Systems*, vol. 25, no. 2, pp. 468-477, February 2014.

[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, February 2013.

[3] M. J. Atallah, M. Blanton, N. Fazio, and K. B. Frikken, "Dynamic and efficient key management for access hierarchies," *ACM Trans. Information and System Security*, vol. 12, no. 3, pp. 18:1-18:43, 2009.

[4] R. S. Sandhu, "Cryptographic implementation of a tree hierarchy for access control," *Information Processing Letters*, vol. 27, no. 2, pp. 95-98, 1988.

[5] F. Guo, Y. Mu, Z. Chen, and L. Xu, "Multi-identity single-key decryption without random oracles," in *Proc. Information Security and Cryptology (Inscrypt '07)*, vol. 4990, pp. 384-398, 2007.

[6] B. Wang, S. S. M. Chow, M. Li, and H. Li, "Storing shared data on the cloud via security-mediator," in *Proc. IEEE 33rd Int'l Conf. Distributed Computing Systems (ICDCS)*, 2013.

[7] 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 *Proc. 10th Int'l Conf. Applied Cryptography and Network Security (ACNS)*, vol. 7341, pp. 526-543, 2012.

[8] D. Boneh, C. Gentry, B. Lynn, and H. Shacham, "Aggregate and verifiably encrypted signatures from bilinear maps," in *Proc. 22nd Int'l Conf. Theory and Applications of Cryptographic Techniques (EUROCRYPT '03)*, pp. 416-432, 2003.

[9] V. Goyal, O. Pandey, A. Sahai, and B. Waters, "Attribute-based encryption for fine-grained access control of encrypted data," in *Proc. of the 13th ACM Conference on Computer and Communications Security (CCS '06)*, pp. 89-98, Alexandria, Virginia, USA, 30 October - 03 November 2006.