International Journal of Scientific Research in Computer Science, Engineering and Information Technology
© 2018 IJSRCSEIT | Volume 4 | Issue 2 | ISSN : 2456-3307

Secure Data distribution with Proxy ReEncryption in Cloud Computing

G.Dhanasekhar[1],Y. Hemalatha[2]
Asst.prof:MCA SVCET,Chittoor, Andhar Pradesh, India
Department of computer applications SVCET,Chittoor, Andhra Pradesh, India

ABSTRACT

Cloud storage services allow users to outsource their data to cloud servers to save local data storage costs. However, unlike using local storage devices, users do not physically manage the data stored on cloud servers However, a way to make sure the cloud user's knowledge security is turning into the most obstacles that hinder cloud computing from extensive adoption. Proxy reencryption is a promising solution to secure the info sharing within the cloud computing. It enables an information owner to write shared data in cloud beneath its own public key, that is any remodeled by a semi trusted cloud server into associate coding meant for the legitimate recipient for access management. This paper offers a solid and provoking survey of proxy reencryption from completely different views to offer a much better understanding of this primitive. Specifically, we reviewed the progressive of the proxy reencryption by investigation the design philosophy, examining the safety models and comparing the potency and security proofs of existing schemes.

Keywords: Cloud storage, security, and encryption.

I. INTRODUCTION

Cloud computing is emerging as a inevitable option for internet based applications and services. Cloud computing is a distributed computing architecture where the computing resources such as hardware, software, processing power are delivered as a service over a network infrastructure. The cloud computing model allows the users to access information and other resources from anywhere that a network connection is available. In cloud computing all data are stored on distributed servers at remote location. The remote locations are data centers. The client can purchase or rent, such as handling time, network bandwidth, disk storage and memory.

Data owners can remotely store their data in the cloud and no longer posses the data locally. Cloud computing migrates the application software and database to the large data centre, where the data management and services may not fully trustworthy. A cloud storage system is a distributed storage system that consists of many independent storage servers. The function of distributed storage systems is to store data confidentially and reliable over long periods of time. The main reason for the raise of the technology cloud computing is because of the convenience that they provide to different newly developed applications and for enterprises. The information that are stored in the cloud is been accessed a huge number of times and is often subjected to changes. An important aspect of cloud storage servers is that, it gives rise to a number of security threats.

Cloud services and applications may require all standard security functions including data confidentiality, integrity, privacy, robustness and access control. Hence securing the cloud and its data is a challenging task. There are several cryptographic methods to secure the data stored in cloud storage

systems. Proxy reencryption is a relatively new data encryption technique devised primarily for distributed data and file security. The target of proxy reencryption is allowing the reencryption of one cipher text to another cipher text without relying or trusting the third party that performs the transfer. In situations where one user wishes for another user to decrypt a message using its own or a new secret key instead of the first user's secret key, one technique involves the assistance of a proxy.

Proxy reencryption is a means for confidential and flexible technique for a user to store and share data. A user can encrypt the file with a public key and then store the ciphertext in a trusted server. When a receiver arrives, the sender can delegate a reencryption key associated with the particular receiver to the trusted server as a proxy. Then the proxy reencrypt the initial ciphertext to the desired receiver.The purpose of proxy reencryption schemes is to prevent the revelation of the keys involved in reencryption and the plaintext that needs to be reencrypted to the proxy.

The Proxy reencryption schemes are basically a version of existing encryption schemes consisting of selection of text, generation of keys, sharing or transmitting of keys between the parties, changeover from plaintext to ciphertext on one end and changeover from ciphertext to plaintext on the other end, the difference arises with the introduction of two more properties Directionality and Transitivity.

Directionality:

If the reencryption scheme is reversible that is, the same reencryption key is used to translate messages from Alice to Bob, as well as from Bob to Alice the scheme is classified as a bidirectional scheme. In these schemes if a user forwards a message to another, it automatically gives rights to the receiver to communicate with the sender. Such reencryption keys are hence generated with the keys in hands of both sender and receiver and with their mutual trust and consent.

A unidirectional scheme is oneway in this context, giving a higher level of security and making it a feasible option in non trusted setups where message conveying is essential but not to an extent where receiver should be given rights to respond to it. So if a message is reencrypted from Alice to Bob with a key, it cannot be used for reencryption from Bob to Alice. Moreover unidirectional schemes are more useful since they can be converted to bidirectional scheme at any time simply by running it in both directions, i.e. from Alice to Bob and from Bob to Alice.

II. PROPOSED ALGORITHM

PROXY REENCRYPTION:

Definition 1 (Proxy ReEncryption):

A proxy reencryption scheme is defined by the following randomized algorithms.

KeyGen: On input the security parameter $k \in K$, the key generation algorithm KeyGen outputs a public/private key pair (pk, sk).

Rekey: On input a key pair (pki, ski) for user i and a key pair (pkj , skj) for user j (skj is optional), the reencryption key generation algorithm Rekey is performed by user i to output a reencryption key $rki \rightarrow j$. In this case, user i act as the delegator and user j acts as the delegate.

Encrypt: On input a plaintext message $m \in M$ and a public key pki for user i, the encryption algorithm Encrypt outputs an original ciphertext $ci \in C1$.

ReEncrypt: On input a ciphertext $ci \in C1$ for user i and a reencryption key $rki \rightarrow j$ for $i \rightarrow j$, the reencryption algorithm ReEncrypt is performed by the proxy to return a transformed ciphertext $cj \in C2$ for user j or the error symbol $\perp$ indicating ci is invalid.

Decrypt: On input a private key ski and a ciphertext $ci \in Cl(l \in \{1, 2\})$ for user i, the decryption algorithm Decrypt is performed by user i to output the corresponding plaintext message $m \in M$ or a error symbol $\perp$ indicating ci is invalid.

Correctness. Typically, the algorithms of KeyGen, Encrypt and Decrypt in PRE scheme are identical to those of normal public key encryption. For any plaintext m ∈ M and two public/private key pairs (pki, ski),(pkj , skj) ←KeyGen(k), the correctness of a proxy reencryption scheme requires that the following equations hold with probability one:

$$\mathsf{Decrypt}(sk_i, \mathsf{Encrypt}(pk_i, m)) = m,$$
$$\mathsf{Decrypt}(sk_j, \mathsf{ReEncrypt}(\mathsf{ReKey}(pk_i, sk_i, pk_j, sk_j), \mathsf{Encrypt}(pk_i, m))) = m.$$

As shown in Figure 1, the aforementioned PRE enables the proxy using a reencryption key rki→j to transform a ciphertext ci for user i under the public key pki into another ciphertext cj for user j under the public key pkj on the same message m ∈ M. Then user j is able to obtain the plaintext message m with his/her private key skj. During the execution of a secure PRE scheme, an attacker (e.g. the proxy) cannot learn any information such as the underlying encrypted message m ∈ M or private keys (e.g. ski or skj).

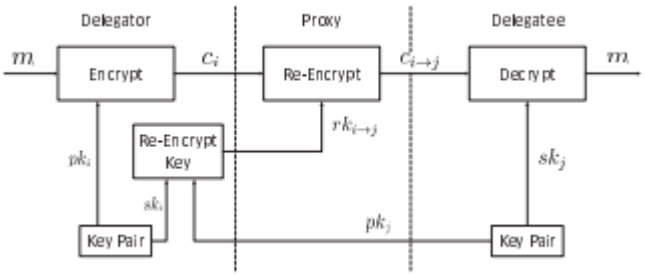

Figure 1. The intuition of Proxy ReEncryption Primitive.

Properties:

To get a sense of the benefits and drawbacks we expect out of a PRE scheme, the most desirable properties of PRE scheme are listed as follows

Unidirectional/Bidirectional:

The PRE scheme is regarded as unidirectional such that the proxy is only allowed to translate the delegator's ciphertext into the delegate's ciphertext on the same message but not vice versa. On the contrary, a bidirectional PRE scheme enables the proxy equipped the reencryption key to transform not only the delegator's ciphertext into the delegate's ciphertext on the same message but also vice versa. One notable difference between the unidirectional and bidirectional PRE scheme relies on the fact that whether the delegate's private key is involved in the reencryption key generation algorithm Rekey or not. Specifically, the bidirectional PRE scheme require that both the delegator (user i) and delegate (user j) must provide their secret keys ski and skj to generate the reencryption key rki→j , whereas only the delegator (user i)'s private key ski is involved to generate the transformation key in the unidirectional PRE scheme.

2) Multiuse/Singleuse:

In a multiuse PRE scheme, ciphertexts generated by either the Encrypt algorithm or ReEncrypt algorithm can be taken as input to reEncrypt to be reencrypted. In contrast, only the original ciphertexts generated by Encrypt can be reencrypted by performing the ReEncrypt algorithm in the singleuse PRE scheme.

3) Keyprivacy:

In a keyprivate PRE scheme, even the proxy performing the translations is unable to disclose the identities of the delegator and delegate from transformation keys or ciphertexts. In other words, the PRE is considered to achieve ciphertext anonymity (a.k.a key privacy) if the malicious proxy and colluding users cannot identify the sender or receiver by observing sufficient reencryption keys or ciphertexts.

4) Transparent: In a transparent PRE, neither the delegator nor the delegate is able to be aware of the existence of the proxy. More formally, it is impossible for any delegate to distinguish an original encryption computed under his public key using the Encrypt algorithm from a reencryption ciphertext on the same message generated by the proxy as the output of the ReEncrypt algorithm. Notably, the input and the corresponding output of the ReEncrypt algorithm in the transparent PRE scheme cannot be linked to each other.

5) Keyoptimal: A user (i.e., the delegator or delegate) is only required to protect and store a small constant number of secret data (i.e., private keys) regardless of how many decryption delegations he/she delegates or accepts. Moreover, the size and number of keys that the proxy is required to safeguard should also remain constant. The purpose of this property is to minimize the safe storage cost for each entity involved in the PRE scheme.

6) Noninteractive: If the secret key skj of the delegate (user j) is not required in the reencryption key generation algorithm rekey, then the underlying PRE scheme is regarded to be noninteractive. That is to say, a reencryption key rki→j can be generated with the delegator's private/public key pair (ski , pkj) and the delegate's public key pkj . I.e., the private key skj of the delegate (user j) is not required as the input of the algorithm Rekey.

7) Nontransitive: The PRE scheme is called to be no transitive if the decryption rights cannot be red legated by the proxy along. Formally speaking, it is infeasible for the proxy to calculate rki→k from rki→j and rkj→k.

8) Temporary: To deal with the case where the delegator needs to revoke the delegated decryption rights, it is desirable to equip the PRE with the temporary property such that the reencryption right for the proxy and the decryption right for the delegate can be deleted according to the request of delegator. It means that the delegator always has power to revoke the delegated right by updating the global parameter or issuing appropriate instructions to the proxy.

9) Collusionresistant: In a collusionresistant PRE scheme, even the proxy colluding with the delegate cannot recover the delegator's private key. Otherwise, the private key of the user will be disclosed in case this user delegates its decryption rights to the malicious proxy and participants. Indeed, there are variant definitions of syntax for PRE schemes, such as the one from Ateniese et al. where sets of Encrypt and Decrypt algorithms instead of single Encrypt and Decrypt algorithms are defined. In this case, these algorithms are defined over different ciphertext spaces (C1 6= C2), where the reencryption function transforms ciphertexts from one space to another, as opposed to the case of a single ciphertext space (C1 = C2), where reencryption maintains the same space. The former syntax is usually associated with multiuse PRE schemes in spite of some recent singleuse schemes are constructed based on the latter syntax. The latter syntax is typically associated with unidirectional PRE schemes where there may be only one direction transformations between ciphertext spaces. It is natural to observe that the syntax of PRE scheme can be adapted dynamically according to the properties featured with the PRE scheme.

III. CONCLUSION

As a promising primitive to secure the information sharing within the cloud computing, PRE has captured lots of concern as a result of the delegation performs of secret writing. During this paper, we tend to review the progressive of the PRE by investigation the planning philosophy, examining the protection models, and examination the potency of existing schemes. The proposed PRE method will provide high efficiency and accuracy.

IV. REFERENCES

[1]. G. Agha. Actors: a model of concurrent computation in distributed systems. MIT Press, Cambridge, MA, USA, 1986.

[2]. M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, and M. Zaharia. A view of cloud computing. Commun. ACM, 53:50–58, April 2010.

[3]. M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. H. Katz, A. Konwinski, G. Lee, D. A. Patterson, A. Rabkin, I. Stoica, and M. Zaharia. Above the clouds: A berkeley view of cloud computing. EECS Department, University of California, Berkeley, Tech. Rep., UCB/EECS-2009-28, Feb 2009.

[4]. C. Clark, K. Fraser, S. Hand, J. G. Hansen, E. Jul, C. Limpach, I. Pratt, and A. Warfield. Live migration of virtual machines. In Proceedings of the 2nd conference on Symposium on Networked Systems Design & Implementation - Volume 2, NSDI'05, pages 273–286, Berkeley, CA, USA, 2005. USENIX Association.

[5]. Hansen, J. Gorm, and E. Jul. Self-migration of operating systems. In Proceedings of the 11th workshop on ACM SIGOPS European workshop, EW 11, New York, NY, USA, 2004. ACM.

[6]. P. Marshall, K. Keahey, and T. Freeman. Improving utilization of infrastructure clouds. In CCGrid 2011, 11th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing, Newport Beach, CA, USA, May 2011.

[7]. B. Rimal, E. Choi, and I. Lumb. A taxonomy and survey of cloud computing systems. In INC, IMS and IDC, 2009. NCM '09. Fifth International Joint Conference on, pages 44 – 51, Aug. 2009.

[8]. C. P. Sapuntzakis, R. Chandra, B. Pfaff, J. Chow, M. S. Lam, and M. Rosenblum. Optimizing the migration of virtual computers. SIGOPS Oper. Syst. Rev., 36:377–390, December 2002.

[9]. J. M. Tirado, D. Higuero, F. Isaila, and J. Carretero. Predictive data grouping and placement for cloud-based elastic server infrastructures. In CCGrid 2011, 11th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing, Newport Beach, CA, USA, May 2011.

[10]. C. Varela and G. Agha. Programming dynamically reconfigurable open systems with SALSA. SIGPLAN Not., 36:20–34, December 2001.

[11]. P. Wang, W. Huang, and C. A. Varela. Impact of virtual machine granularity on cloud computing workloads performance. In Workshop on Autonomic Computational Science (ACS'2010), Brussels, Belgium, October 2010.

[12]. L. Wu, S. K. Garg, and R. Buyya. SLA-based resource allocation for software as a service provider (SaaS) in cloud computing environments. In CCGrid 2011, 11th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing, Newport Beach, CA, USA, May 2011.

[13]. M. Armbrust, A. Fox, R. Griffith, A.D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica and M. Zaharia,"A View of Cloud Computing," Commun. ACM, vol. 53, no. 4, pp. 50-58, 2010.

[14]. R.Buyya, C.S.Yeo, S.Venugopal, J.Broberg and I.Brandic,"Cloud Computing and Emerging It Platforms: Vision, Hype, and Reality for Delivering Computing as the 5th Utility," Fut. Gener. Comput. Syst., vol. 25, no. 6, pp. 599-616, 2009.

[15]. L.Wang, J. Zhan, W. Shi and Y. Liang,"In Cloud, Can Scientific Communities Benefit from the Economies of Scale?," IEEE Trans. Parallel Distrib. Syst., vol. 23, no. 2, pp. 296-303, 2012.

[16]. H. Takabi, J.B.D. Joshi and G. Ahn,"Security and Privacy Callenges in Cloud Computing Environments," IEEE Security & Privacy, vol. 8, no. 6, pp. 24-31, 2010.

[17]. D. Zissis and D. Lekkas,"Addressing Cloud Computing Security Issues," Fut. Gener. Comput. Syst., vol. 28, no. 3, pp. 583- 592, 2011.

[18]. D. Yuan, Y. Yang, X. Liu and J. Chen,"On-Demand Minimum Cost Benchmarking for Intermediate Dataset Storage in Scientific Cloud Workflow Systems," J. Parallel Distrib. Comput., vol. 71, no. 2, pp. 316-332, 2011.

[19]. S.Y. KO, I. Hoque, B. Cho and I. Gupta,"Making Cloud Intermediate Data Fault-Tolerant," Proc. 1st ACM Symp. Cloud Computing (SoCC'10), pp. 181-192, 2010.