Secure Cloud System for Robust and Highly Scalable Storage, Retrieval and Forwarding with Cooperation

B Sai Sindhuri, A Sandhya Rani, P.V.S. Srinivas

Abstract: Cloud computing is a technology that facilitates the sharing of computing resources in pay per use fashion. Especially it can be used to store huge amount of data. The data of cloud users is stored in multiple servers at the service provider. The servers are considered untrusted. For this reason there are security concerns among the people for outsourcing their valuable business data to cloud. In existing system encryption techniques are used. However, they are proved to cause much overhead on cloud storage. Moreover encryption techniques cause many restrictions on data dynamics. Therefore there is a need for securing cloud data and also support data dynamics. Towards this many techniques came into existence. We proposed a scheme known as threshold proxy re-encryption along with erasure codes to solve the problem. They used security servers and key servers for storing data and keys respectively. This scheme is efficient is secure storage, retrieval and forwarding of data. However, assuming inconsistencies among servers, this paper proposes a timestamp based solution that extends the scheme for more robust security for cloud storage. The experimental results revealed that the proposed solution is effective and can be used in real clouds.

Keywords: Threshold Proxy re-encryption scheme, secure cloud storage system, storage key servers, Timestamp approach

I. INTRODUCTION

Many technical innovations changed the way computing takes place. The technologies like cloud computing and virtualization brought a new phenomenon or model of computing. This model allows individuals and organizations to access huge amount of computing resources. Moreover this is possible in pay as use fashion without investment. The cloud computing technology brought this change in the computing world. Cloud computing became a reality. Many cloud service providers such as IBM, Microsoft, Amazon, Google and son on are providing cloud services. The cloud services are available in the form of Platform as a Service (PaaS), Software as a Service (SaaS) and Infrastructure as a Service (IaaS). These services changed the way computing and storage takes place. The technology was made viable due to the success of virtualization. Cloud infrastructure can store huge amount of data. However, there are security concerns over cloud storage. For this reason people do not shift to using cloud as faster as expected. Lin et al. [1] focused on solving this problem. They have developed a security scheme as presented in fig. 1. This scheme supports data dynamics with complete security. Towards improving robustness of cloud storage, many techniques came into existence. One of them is to replicate the data among many servers. Erasure codes technique is another way of securely storing data. Erasure codes can recover lost data. However, this technique has tradeoffs between the storage size and threshold of failures. For this reason a decentralized erasure code is desired towards reliable cloud storage.

Data confidentiality is an important feature desired of cloud storage. The cryptographic methods used to ensure confidentiality have exhibited tradeoffs with data dynamics. In this approach communication cost is high in the network. Moreover managing crypto keys is another problem. In the scheme of Lin et al. security is lost when key server is comprised. Yet another problem of such schemes is that servers do not support secure data forwarding. This paper proposes a new scheme to solve this problem which is an extension to the one proposed by Lin et al. It focuses on security concerns due to communication inconsistencies among the cloud servers. The storage servers and key servers work together. Communication among them is to be consistent for data integrity and security. This paper assumes that there might be serious inconsistencies in communication among the servers. As storing data keys in single server is not safe, multiple storage servers and multiple key servers are used in the scheme proposed by Lin et al. [1]. This paper uses the same scheme with timestamp based mechanism to prevent the inconsistencies. Thus the threshold proxy re-encryption scheme is enhanced to have more robust cloud storage security. This is achieved through perfect cooperation among the servers through timestamp based solution.

The remainder of this paper is organized as follows. Section II reviews literature on cloud storage security. Section III provides the overview of the proposed system which enhances Lin et al.'s scheme. Section IV provides details of implementation and evaluation. Section V presents experimental results while section VI concludes this paper.

II. RELATED WORK

- *B Sai Sindhuri is currently pursuing masters degree program in Computer Science engineering in TKR College of Engineering, Hyderabad - 500097, INDIA PH-+91-9966178190. E-mail: Sindhuri.sai7374@gmail.com*
- *A Sandhya Rani, Assistant Professor, Department of Computer Science and Engineering, TKR College of Engineering and Technology, Hyderabad, A.P-500 097, India*
- *P.V.S. Srinivas, Professor& Head, Department of Computer Science and Engineering, TKR College of Engineering and Technology, Hyderabad, A.P-500 097, India*

Cloud storage needs distributed file system. Many file systems were proposed. They include Network File System [2], Network Attached Storage [3] etc. These file systems are scalable and they are decentralized and distributed in nature. Replica management and other techniques were invented to make the distributed file systems secure and robust. Later on more improvements were proposed in [4], [5] to add features to file system such as scalability and efficiency. Lot of research was into erasure codes [6], [7], [8], [9] and [10] used for storage security in distributed environments. These techniques convert the data into code words. The code word is a vector of symbols. These symbols are used to represent storage problems. Data is stored in multiple servers. While retrieving data the data from multiple servers is combined and returned. Thus the probability of retrieval and security are high while communication cost is higher. Nevertheless, when storage servers are compromised, data confidentiality is not guaranteed. Lin and Tzeng [11] addressed this problem using erasure codes. Proxy re-encryption schemes are used in [12] and [13]. This scheme supports secure data forwarding using public key encryption. In [14] another such technique is used which is based on sharing function. According to this technique user sends re-encryption key to server when he wants to share data. The storage server re-encrypts the already encrypted data for confidentiality. A type - based proxy re-encryption scheme is presented by Tang [15] which has more control over re-encryption key. It allows users to choose message before the scheme is applied. Key-private proxy re-encryption [16] is another such scheme where the identity of the recipient is kept confidential. Pairing concept is missing all the above schemes [17]. Integrity verification schemes [18], [19] and [20] are proposed for secure storage, retrieval and forwarding. The notion of provable data possession is used in [21] and [22]. These techniques use messages in plain text. Lin et al. [1] focused on the cloud storage security with respect to data storage retrieval and forwarding. This paper improves their scheme by implementing a timestamp based mechanism that ensures perfect cooperation among the servers. Thus the problem of communication inconsistencies among the servers is addressed.

III. THE PROPOSED SYSTEM

The proposed timestamp based scheme is an extension to the scheme proposed by Lin et al. The overview of the scheme is presented in fig. 1. It is used for secure cloud storage, data retrieval and data forwarding. The concepts like encryption, erasure codes, and proxy re-encryption concepts are used for cloud storage security.

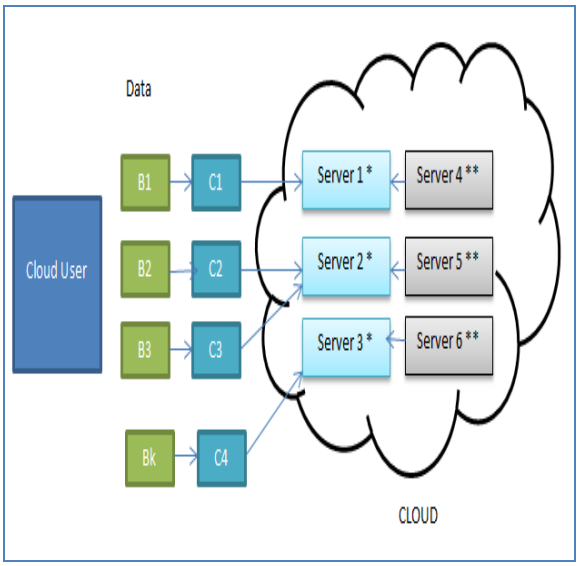

Fig. 1 – Overview of Proposed Scheme

As shown in fig. 1, there are storage servers (denoted by *) and key servers (denoted by **) in the cloud. The data of cloud users is stored in multiple servers. The corresponding security keys are stored in key servers. The functionality of the proposed scheme is understood in terms of setup, data storage, data retrieval and data forwarding with integrity.

Setup

This is the first phase in which system managers sets required parameters. Afterwards, every user is assigned a pair of keys as part of public-key cryptography. Afterwards, the user's secret key is stored in key server.

Data Storage

This phase is meant for secure storage of data. Cloud users perform this activity. When a cloud user wants to outsource a file to cloud, he will break it into some blocks. Then each block is encrypted. The encrypted blocks are saved to multiple storage servers of the cloud. The servers receive cipher text and convert them into code words and store them. There are two important operations involved in the storage process. They are computing the identity token, invoking encryption algorithm and encoding. The computation of token is done as follows.

$$\tau = h^{f(a_3,\mathrm{ID})}$$

Then the encryption algorithm takes place as follows.

$$C_i = (0, \alpha_i, \beta, \gamma_i) = (0, g^{r_i}, \tau, m_i \tilde{e}(g^{a_1}, \tau^{r_i})),$$

Afterwards encoding process takes place which is responsible to generate code words for the content which is in the form of ciphertext. The encoding process is performed as follows.

$$C' = \left(0, \prod_{i=1}^{k}(\alpha_i^{g_i}), \beta, \prod_{i=1}^{k}(\gamma_i^{g_i})\right)$$
$$= \left(0, g^{\sum_{i=1}^{k} g_i r_i}, \tau, \prod_{i=1}^{k} m_i^{g_i} \tilde{e}(g^{a_1}, \tau)^{\sum_{i=1}^{k} g_i r_i}\right)$$
$$= (0, g^{r'}, \tau, W\tilde{e}(g, \tau)^{a_1 r'}),$$

Data Forwarding

Users of cloud can forward their data to other users. It takes place with public key cryptography. When user A wants to send data to user B, the user A encrypts data with public key of user B and send the data. On receiving data the user B can decrypt it with his own private key. The data forwarding is done through servers. Before data is forwarded to recipient the servers re-encrypt the data using public key of B. Then the data is forwarded to B. There are three algorithms involved in the data forwarding process. They are known as KeyRecover(.), ReKeyGen(.), and ReEnc(.). When user needs first component's secret key KeyRecover(.) algorithm in invoked. It is performed as follows.

$$a_1 = \sum_{s \in \mathrm{T}} \left(f_{\mathrm{A},1}(s) \prod_{s' \in \mathrm{T}/\{s\}} \frac{-s'}{s - s'} \right) \bmod p.$$

For generating re-encryption key, the ReKeyGen(.) is invoked. This algorithm in turn invoke Re-Enc(.) algorithm. The-ReKeyGen(.) is performed as follows.

$$\mathrm{RK}_{\mathrm{A}\to\mathrm{B}}^{\mathrm{ID}} = \left((h^{b_2})^{a_1(f(a_3,\mathrm{ID})+e)}, h^{a_1 e}\right).$$

For generating re-encrypted codeword symbols, the ReEnc(.) algorithm performs the following.

$$C'' = \left(1, \alpha, h^{b_2 a_1 (f(a_3,\mathrm{ID})+e)}, \gamma \cdot \tilde{e}(\alpha, h^{a_1 e})\right)$$
$$= \left(1, g^{r'}, h^{b_2 a_1 (f(a_3,\mathrm{ID})+e)}, W\tilde{e}(g, h)^{a_1 r'(f(a_3,\mathrm{ID})+e)}\right).$$

Data Retrieval

It is a process of retrieving data with complete integrity. When a user sends data retrieval request, the user is authenticated by key servers. Then the storage servers do partial decryption of the available data and then combine the whole data before sending it to the cloud user. More details on data storage, retrieval and forwarding can be found in [1]. ShareDoc(.) and Combine(.) are the two algorithms involved in data retrieval. ShareDoc(.) is invoked by a key server after obtaining original codeword symbols in order to perform partial decryption. Then the partial decryption takes place at multiple servers where pieces of data are stored. Then Combine(.) algorithm is invoked to club all the pieces to get original file.

Fault Tolerance

Fault tolerance is built into the framework of the cloud infrastructure. When a server is down for any reason, the other servers will continue processing requests. Later on steps can be taken to recover the server which failed to process request.

IV. IMPLEMENTATION AND EVALUATION

Implementation of the proposed scheme is as follows. The implementation of setup, secure cloud storage, data forwarding and data retrieval are similar as explored in [1]. This paper focuses on timestamp-based cooperation among the key servers and storage servers. This cooperation among the servers ensures consistency in data dynamics. Data consistency and fair handling of request are considered in the proposed scheme. The data inconsistencies due to communication delays in the existing systems are overcome here using timestamp – based mechanism. The new scheme uses a global time stamp which is followed by storage and key servers. As many servers are involved in the operations, the data dynamics are to be carried out with consistency and security.

All operations such as data storage, data retrieval and data forwarding are to be taken place with integrity. For instance when users send data to cloud, the storage process involves multiple servers. The timestamp based solution monitors the transaction and ensures that perfect storage takes place as expected. In case of communication concerns, the new technique has to take steps to ensure consistency. This approach is followed in data retrieval and data forwarding also. We built a prototype application for testing the efficiency of the proposed solution. The experimental results revealed that the timestamp based solution can prevent inconsistencies in cloud storage.

In fact in all operations such as Enc, Encode, KeyRecover, ReKeyGen, ReEnc, ShareDec and Combine, a timestamp is associated for integrity of operations associated with a single transaction. The timestamp is somehow related to the ID of the present transaction. The aim of the timestamp-based operations is to ensure that all operations in a single transaction, where multiple servers are involved, are executed as a unit. Thus more cooperation and robust integrity of the operations can be achieved.

V. EXPERIMENTAL RESULTS

We have made experiments in custom simulator built in Java platform. The cloud servers, cloud server providers and the data owners, the operations involved are simulated. The simulation results reveal that the proposed timestamp approach outperforms the existing approach.

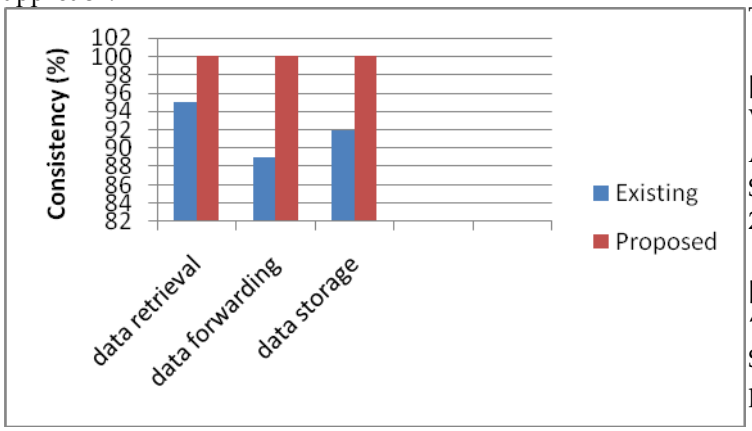

Fig 2 Comparison of consistency

As shown in the above figure 2 represents horizontal axis is the data retrieval, forwarding and storage while vertical axis represents consistency.

VI. CONCLUSION

In this paper we present a new timestamp-based scheme for cloud storage security. The new scheme is built on existing one for robust data storage, data retrieval and data forwarding. When data is stored and retried, multiple servers are involved in cloud. This paper focused on addressing communication concerns among the storage and key servers in the cloud. The solution is through timestamp based mechanism that ensures data integrity in all operations.

VII. REFERENCES

[1] Hsiao-Ying Lin, Member, IEEE, and Wen-GueyTzeng, Member, IEEE, "A Secure Erasure Code-Based Cloud Storage System with Secure Data Forwarding", IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, VOL. 23, NO. 6, JUNE 2012.

[2] P. Druschel and A. Rowstron, "PAST: A Large-Scale, Persistent Peer-to-Peer Storage Utility," Proc. Eighth Workshop Hot Topics in Operating System (HotOS VIII), pp. 75-80, 2001.

[3] M. Kallahalla, E. Riedel, R. Swaminathan, Q. Wang, and K. Fu, "Plutus: Scalable Secure File Sharing on Untrusted Storage," Proc. Second USENIX Conf. File and Storage Technologies (FAST), pp. 29- 42, 2003.

[4] Z. Wilcox-O'Hearn and B. Warner, "Tahoe: The Least-Authority Filesystem," Proc. Fourth ACM Int'l Workshop Storage Security and Survivability (StorageSS), pp. 21-26, 2008.

[5] S.C. Rhea, P.R. Eaton, D. Geels, H. Weatherspoon, B.Y. Zhao, and J. Kubiatowicz, "Pond: The Oceanstore Prototype," Proc. Second USENIX Conf. File and Storage Technologies (FAST), pp. 1-14, 2003.

[6] R. Bhagwan, K. Tati, Y.-C. Cheng, S. Savage, and G.M. Voelker, "Total Recall: System Support for Automated Availability Management," Proc. First Symp.Networked Systems Design and Implementation (NSDI), pp. 337-350, 2004.

[7] A.G. Dimakis, V. Prabhakaran, and K. Ramchandran, "Decentralized Erasure Codes for Distributed Networked Storage," IEEE Trans. Information Theory, vol. 52, no. 6 pp. 2809-2816, June 2006.

[8] H.-Y. Lin and W.-G.Tzeng, "A Secure Decentralized Erasure Code for Distributed Network Storage," IEEE Trans. Parallel and Distributed Systems, vol. 21, no. 11, pp. 1586-1594, Nov. 2010.

[9] M. Mambo and E. Okamoto, "Proxy Cryptosystems: Delegation of the Power to Decrypt Ciphertexts," IEICE Trans. Fundamentals of Electronics, Comm. and Computer Sciences, vol. E80-A, no. 1, pp. 54- 63, 1997.

[10] M. Blaze, G. Bleumer, and M. Strauss, "Divertible Protocols and Atomic Proxy Cryptography," Proc. Int'l Conf. Theory and Applicationof Cryptographic Techniques (EUROCRYPT), pp. 127-144, 1998.

[11] 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.

[12] Q. Tang, "Type-Based Proxy Re-Encryption and Its Construction," Proc. Ninth Int'l Conf. Cryptology in India: Progress in Cryptology (INDOCRYPT), pp. 130-144, 2008.

[13] J. Shao and Z. Cao, "CCA-Secure Proxy Re-Encryption without Pairings," Proc. 12th Int'l Conf. Practice and Theory in Public Key Cryptography (PKC), pp. 357-376, 2009.

[14] H. Shacham and B. Waters, "Compact Proofs of Retrievability," Proc. 14th Int'l Conf. Theory and

Application of Cryptology and Information Security
(ASIACRYPT), pp. 90-107, 2008