A Secure Cloud Storage using Erasure Code Mechanism with Secure Data Forwarding

N.Rajeswari[1], G.Manisha[1], S.Venkatalakshmi[2]
[1]B.E-Final Year, [2]Assistant Professor
[1,2]Department of Computer Science and Engineering, K.L.N. College of Engineering
[1,2]Sivagangai District, Tamilnadu, India

Abstract **- Cloud storage consists distributed storage where digital data is stored in logical pools. Although cloud storage has many advantages like data scalability, cost savings, usability, the major threat to cloud storage is data security. General encryption schemes limit few operations on encrypted data. Data robustness is a major requirement in distributed storage system. We propose a secure cloud storage which ensures data robustness by using decentralized erasure code mechanism. Threshold Proxy re-encryption scheme is used where a user can forward the data in storage servers to another user without retrieving it back. This scheme proposes encoding and forwarding operations over the encrypted data.**

Keywords—Decentralized Erasure Code, Threshold Proxy Re-Encryption, Security.

I. INTRODUCTION

A cloud storage is considered as a large scale distributed storage system that consists of many independent storage servers. There have been many ways of storing data over storage servers. Data robustness is provided by replicating a message such that each storage server stores a copy of the message. This is very robust because the message can be retrieved as long as one storage server survives. Erasure coding encodes a message of k symbols into a codeword of n symbols. To store a message, each of its codeword symbols is stored in different storage server. Storage server failure leads to an erasure error of the codeword symbol. As long as the number of failure servers is under the tolerance threshold of the erasure code, the message can be retrieved from the codeword symbols stored in the available storage servers by the decoding process. Decentralized erasure code computes each codeword symbol for a message independently. Thus, the encoding process for a message can be split into n parallel tasks of generating codeword symbols. The message symbols are then sent to the storage servers. Upon receiving it, each storage server computes a codeword symbol for the message symbols and stores it. Thus the encoding and storing process is completed. The recovery process is the same. There is a serious concern on data confidentiality when storing our data in third party's cloud server . To provide strong confidentiality for messages in storage servers, a user can encrypt messages by a cryptographic algorithm before applying an erasure code method to encode and store messages. When he wants a message, he needs to retrieve the codeword symbols from storage servers in which they are stored. It is then decoded and decrypted using cryptographic keys.

II. RELATED WORKS

We briefly review parallel and distributed storage systems, proxy re-encryption schemes, and integrity checking mechanisms.

A. Parallel and Distributed Storage Systems

Parallel and distributive storage systems have undergone impressive change over recent years. New architectures and applications have fast become the central focus of the development. These changes are a result of cross fertilization of parallel and distributed technologies with other fast evolving technologies. It is of paramount importance to review and assess these new developments in contrast with new research achievements in the well-established areas of parelle and distributing. At the early years, the network attached storage and the Network File Systems (NFS) offers extra storage devices over the network such that a user can access the storage devices using network connection. Many improvements on scalability, robustness, and security were proposed [1]. A storage system with decentralized architecture provides good scalability, since a storage server can join or leave without the permission of a pivotal authority. A simple method to provide robustness against server failures is to make replicas of each message and store them in different servers.

However, this approach is expensive as Z replicas bring about Z times of expansion. One way to reduce the expansion rate is to use erasure codes to encode messages [3]. A message is encoded as codeword, which is a vector of symbols, and each storage server stores a codeword symbol. A storage server failure is as an erasure error of the stored codeword symbols. Random linear codes support distributed encoding which means each codeword symbol is independently computed. To store a message of k blocks, each storage server linearly incorporates the blocks with randomly chosen coefficients. Then both the codeword symbol and coefficients are stored. To retrieve the message, a user queries k storage servers for the stored codeword symbols and coefficients. And then the linear system is solved.

B. Proxy Re-Encryption schemes

Proxy re-encryption (PRE) allows a semi-trusted proxy to convert a cipher-text intended for a user into a cipher text for another user without knowing about the hidden plaintext. Chunbo Ma et al. have proposed a group based proxy re-encryption scheme to convert a cipher text from one group to another. Any group member can independently decrypt the cipher texts encrypted to its group. Proxy re-encryption schemes are proposed by Mambo and Okamoto[14] and Blaze et al.[15].A proxy server can convert a cipher text under a public key of user A (PUKA) to a new one under another public key of user B (PUKB) by using the re-encryption key RK A→B. The server does not know the plaintext during conversion because the messages are first encrypted by the owner. Type based proxy re-encryption schemes [4] gives a better granularity on the granted right of a re- encryption key. A user can decide which type of messages and with whom he wants to share in this type of scheme. Key private proxy re-encryption schemes are proposed by Ateniese et al. [18]. Given a re-encryption key in a proxy re-encryption scheme, a proxy server will not be able to determine the identity of the recipient. This kind of proxy re-encryption schemes adds higher privacy against proxy servers. Even though most proxy re-encryption schemes make use of pairing operations, there are proxy re-encryption schemes without pairing [19].

C. Integrity Checking Functionality

Another important functionality about cloud storage is the function of integrity checking. Once a user stores data into the storage system, he no longer has the data at hand. The user may want to verify whether the data are properly stored in storage servers. The concepts of provable data possession [20], [21] and the proof of storage [22], [23], [24] were proposed. Then public auditability of stored data is addressed in [25].

III.SYSTEM MODEL

Our system model consists of users, n storage servers S1;S2;.....;Sn and m key servers K1;K2;.....;Km. Storage servers offers storage services and key servers provide key management services. They work independently. Our distributed storage system comprises of four phases: system setup, data storage, data forwarding and data retrieval. These four phases are described as follows.

In the system setup phase, the system manager selects the system parameters and publishes them. Every user A is assigned a public-secret key pair PUKA; SEKA. User A distributes his secret key SEKA to key servers such that each key server Ki holds a key share SEKA. The key is shared with a threshold t.

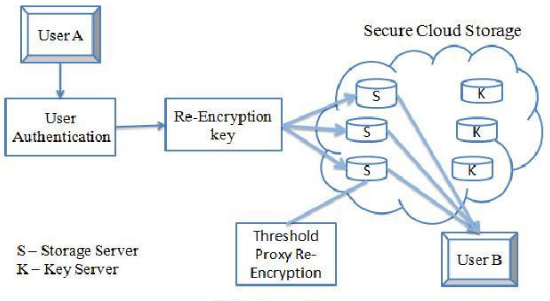
In data storage phase, user A encrypts his message M and dispatches it to storage servers. A message M is decomposed into k blocks m1;m2;.....;mk and has an identifier ID. User A encrypts each block mi into cipher text Ci and sends it to v randomly chosen storage servers. Upon receiving cipher texts from a user, each storage server linearly incorporates them with randomly chosen coefficients into a code word symbol and stores it.

A storage server may receive less than k message blocks and we assume that all storage servers know the value k in advance.

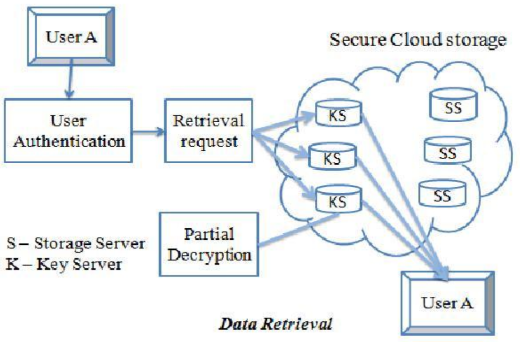
In the data forwarding phase, A forwards his encrypted message with identifier ID stored in storage servers to user B so that B can decrypt the forwarded message by his secret key, To achieve this, A uses his secret key (SEKA) and B’s public key (PUKB) to compute a re-encryption key RKID A!B and then sends this key to all storage servers. Upon receiving the key, each storage server uses the re-encryption key to re-encrypt its code word symbol. The re-encrypted code word symbol is the combination of cipher texts under B’s public key

PUKB. To distinguish re-encrypted code word symbols from untouched ones, we call them original code word symbols and re-encrypted code word symbols, respectively.

In the data retrieval phase, user A requests to fetch a message from storage servers. The message of user is either stored by him or forwarded to him. User A sends a retrieval request to key servers. Upon receiving the request and executing a appropriate authentication process with user A, each key server Ki requests randomly chosen storage servers to get codeword symbols and performs partial decryption on the received codeword symbols by using the key share SEKA;i. Finally, user A combines the partially decrypted codeword symbols to obtain the original message M. When a storage server fails, a new one is added. The new storage server queries k available storage servers, linearly combines the received code word symbols as a new one and stores it. The system is then recovered.



Data Forwarding



Data Retrieval

A Straightforward Solution-

A straightforward solution for supporting the data forwarding in a distributed storage system is as follows: When the owner A wants to send a message to user B, he downloads the encrypted message and decrypts it by using his secret key. Then he encrypts the message by using B’s public key and uploads the new cipher text. When user B wants to retrieve the forwarded message from user A, he downloads the cipher text and then decrypts it by his secret key. The full data forwarding process needs three communication processes for

A’s downloading and uploading and then B’s downloading. The communication cost is linear in the length of the forwarded message. The communication cost is linear in the length of the forwarded message. The decryption and encryption for the owner A, and the decryption for user B is the computation cost. Proxy re-encryption schemes can significantly decrease communication and computation cost of the owner. In a proxy r-encryption scheme, the owner sends a re-encryption to storage servers such that the storage servers perform the re-encryption operation for him. Thus, the communication cost of the owner does not depend on the length of forwarded message and the computation cost of re-encryption is taken care of by storage servers. In a secure storage system, Proxy re-encryption schemes significantly reduce the overhead of the data forwarding function.

IV.METHODOLOGY

A. Erasure Code Mechanism

Consider two cyclic multiplicative groups G1 and G2.Assign a prime order p and g.Encoder generates a generator matrix **G=[$g_{i,j}$]** for 1<i<k,1<j<n.

For each row, the encoder randomly selects an entry and randomly sets a value from Z_P^* to the entry.The encoder repeats this step v times with replacement for each row.An entry of a row can be selected multiple times but only set to one value.The values of the rest entries are set to 0.Message is considered as(m_1,m_2,….m_k) €

G_2^K.Encoding process is to generate (w_1,w_2,…w_n) € G_2^N.

Decoding process is to compute inverse of a k*k submatrix K of G.

$$K^{-1}=[d_{i,j}]$$

Final step of decoding process is to compute $m_i=w_{j1}^{d1},w_{j2}^{d2},\dots w_{jk}^{dk}$.When storage servers SS_1 and SS_3 are available and k*k submatrix K is invertible. Then user A can decode m1 and m2 from the codeword symbols w_1,w_3 and the coefficients(g1,1,0),(0,g2,3),which are stored in the storage servers SS_1 and SS_3.

B.Proxy Re-Encryption Mechanism

User A wants to forward a message to another user B. He needs the first component a1 of his secret key. If A does not possess a1, A queries key servers for key shares. A can recover the first component a1 of the secret key when atleast t key servers respond to it.

Let the identifier of the message be ID. User A computes the re-encryption key RK and securely sends the re-encryption key to each storage server.

By using RK, a storage server re-encrypts the original codeword symbol C’ with the identifier ID into a re-encrypted codeword symbol C’’ such that C’’ is decrypted using B’s secret key.

A threshold proxy re-encryption scheme has multiplicative homomorphic property. An encryption scheme is multiplicative homomorphic
if it supports a group operation ⊙ on encrypted laintexts without decryption

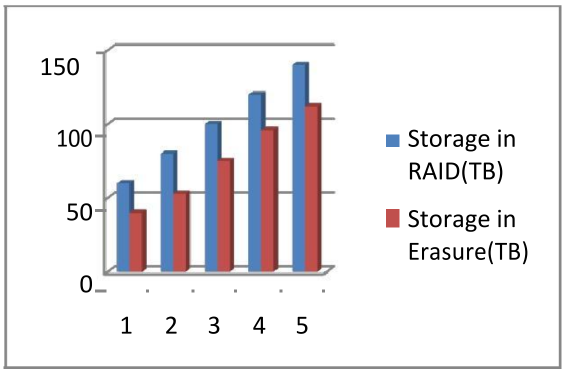
$$D (SEK, E(PUK, m_1) \odot E (PUK, m_2)) = m_1 . m_2$$

where E is the encryption function, D is the decryption function, and (PUK,SEK) is a pair of public key and secret key.

V.PERFORMANCE EVALUATION

Many features like De-duplication, Compression, Thin Provisioning, Snapshots, Clones and RAID were implemented to make storage more efficient. In case of RAID, disk drives have become larger but their reliability has stayed the same. Recovering a failed disc takes more time; it is reconstructed using RAID parity information. There is also increasing probability of a second disk failure or other errors before reconstruction can complete. It’s not uncommon for a 4TB disk to take days to rebuild.

Implementation of Erasure Coding overcomes works by creating a mathematical function around a data set such that if a member of the data set is lost, the lost data can be recovered easily from the rest of the members of the set. Common industry uses Erasure coding for storage efficiency.



VI.CONCLUSION AND FUTURE WORKS

In this paper, we consider a cloud storage system consists of storage servers and key servers. We integrate a newly proposed threshold proxy re-encryption scheme and decentralized erasure codes. The threshold proxy re-encryption scheme enhances encoding, forwarding, and partial decryption operations in a distributed way. To decrypt a message that are encrypted and encoded to codeword symbols, each key server only has to partially decrypt two codeword symbols in our system. By using the threshold proxy re-encryption scheme, we present a secure cloud storage system that provides secure data storage by erasure code mechanism with secure data forwarding functionality. Each storage server independently performs encoding and re-encryption and partial decryption is performed by independent key servers. Our storage system and some newly proposed content addressable file systems and storage system are highly compatible. The storage servers which are storage nodes in a content addressable storage system are for storing content addressable blocks. Our key servers act as access nodes for providing a front-end layer such as traditional file system interface. We further enhance more flexibility in the implementation of Erasure Coding mechanism.

VII.ACKNOWLEDGEMENT

We express our gratitude to anonymous, untraceable reviewers and committee members of the conference for their very helpful comments. More details of this design are available in our technical report.

REFERENCES

[1] J. Kubiatowicz, D. Bindel, Y. Chen, P. Eaton, D. Geels, R. Gumadi, S. Rhea, H. Weatherspoon, W. Weimer, C. Wells and B. Zhao," Oceanstore: An Architecture for Global-Scale Persistent Storage," Proc. NinthInt' l Conf. Architectural support for programming languages and operating systems(ASPLOS), pp.190-201,2000.

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

[3] Adya, W. J. Bolosky, M. Castro, G. Cermak, R. Chaiken, J. R. Douceur, J. Howell, J. R. Lorch, M. Theimer, and R.Wattenhofer," Farsite:Federated, Available, and Reliable Storage for an Incompletely Trusted Environment," Proc.Fifth Symp. Operating

[4] Haeberlen, A. Mislove, and P. Druschel, "Glacier: HighlyDurable, Decentralized Storage Despite Massive Correlated Failures,"Proc. Second Symp.Networked Systems Design and implementation(NSDI),pp.143-158,2005.

[5] 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.

[6] 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.

[7] D. R. Brownbridge, L. F. Marshall, and B. Randell, "The Newcastle Connection or Unixes of the World Unite!,"Software Practice and Experience, vol. 12, no. 12,pp. 1147-1162,1982.

[8] R.Sandberg, D.Goldberg, S.Kleiman, D.Walsh, and B.Lyon,"Design and Implementation of the Sun Network Filesystem," Proc.USENIX Assoc. Conf.1985.

[9] 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.

[10] 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.

[11] 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.

[12] A. G. Dimakis, V. Prabhakaran, and K.Ramchandran, "Ubiquitous Access to Distributed Data in Large-Scale Sensor Networks through Decentralized Erasure Codes," Proc. Fourth Int'l Symp. Information Processing in Sensor Networks (IPSN), pp. 111-117,2005.

[13] A. G. Dimakis, V. Prabhakaran, and K. Ramchandran, "Decentralized Erasure codes