ISSN: 2277-9655
Scientific Journal Impact Factor: 3.449
(ISRA), Impact Factor: 2.114

IJESRT
INTERNATIONAL JOURNAL OF ENGINEERING SCIENCES & RESEARCH TECHNOLOGY

Re-Encryption Scheme to Provide Secure Data Forwarding in Cloud Storage System

D. Hema Latha*, D. Rama Krishna Reddy, K. Sudha
*Asst. Professor, Dept of Computer Science, Osmania University College for Women, Koti, Hyderabad
Asst. Professor in Computer Science, Dept of Mathematics, UCS, Osmania University, Hyderabad

Abstracts

A cloud storage system is considered as a large scale distributed storage system which consists of several independent storage servers. In Cloud computing the resources on the Internet are considered and managed as a combined entity called cloud. Data stored in the third party cloud system might create serious concern on data confidentiality. To provide strong confidentiality for messages and information in storage servers, the user can encrypt messages by a cryptographic method. Most conventional encryption schemes protect data confidentiality but also limit the functionality of the storage system as only few operations are supported over encrypted data.

In this paper, the authors discussed about the problem of forwarding data to another user by storage servers directly under the command of data owner. To provide a secured data forwarding, we propose a proxy re-encryption scheme and integrate it with a secured decentralized code to form secure distributed storage system. The constricted integration of encoding, encryption and data forwarding build the storage system efficiently and effectively meet the specifications of data robustness, data confidentiality, and data forwarding.

Keywords: Cloud storage system; Decentralized code; proxy re-encryption; cryptography; secure storage system; secret key; encryption key; key server..

Introduction

Cloud storage is a model of network enterprise storage where data is stored in virtualized pools of storage that are generally hosted by third parties. Large data centers are maintained and operated by hosting companies and clients or people who need their data to be hosted, purchase or lease storage capacity hosting companies. In the background, the data center operators virtualizes the resources according to the prerequisite or necessity of the customer and display them as storage pools, and these storage pools can be used by the customers themselves to store files or data objects. The resource may spread across multiple servers and multiple locations. The safety of the files is the responsibility of the hosting companies, and on the applications that influence the cloud storage.

High-speed networks and universal Internet access is available in recent years, so many services are provided on the Internet so that the users can use them from anywhere at any time. In this paper, we concentrate on design issue of cloud storage system for confidentiality, robustness, operative and serviceable. A cloud storage system is treated as a large scale distributed storage system that is comprised of many independent storage servers.

Data robustness is a main necessity for storage systems. Numerous proposals have been made for storing data over storage servers [1]. To provide data robustness, encode a message of k symbols into a codeword of n symbols by erasure coding method [2]. To store a message, each and every codeword symbol is buffered in a separate storage server or decentralized storage server [3]. A decentralized erasure code is an erasure code which computes each codeword symbol for a message independently.

Thus, the encoding process for a message can be divided into ‘n’ parallel tasks of generating codeword symbols. In distributed storage system, a decentralized erasure code is well suited. After the message symbols are sent to storage servers, each storage server independently calculates a code word symbol for the received message symbols and stores it. This is the encoding and storing process. The retrieval process is the same. The working of cloud computing is shown in Fig. 1. The cloud computing network structure [4] is shown in Fig. 2. The cloud storage with multiple devices is shown in Fig. 3.

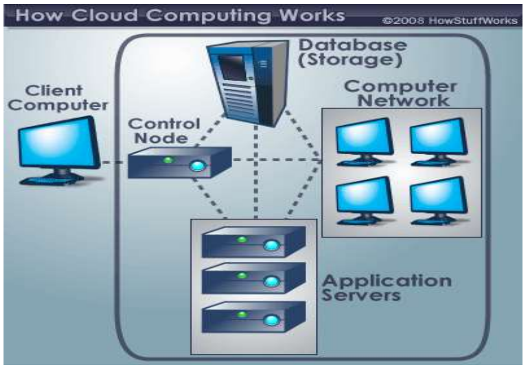

Fig. 1. Cloud computing work

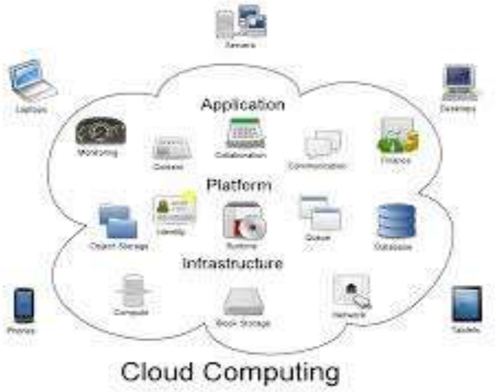

Fig. 2. Cloud computing network

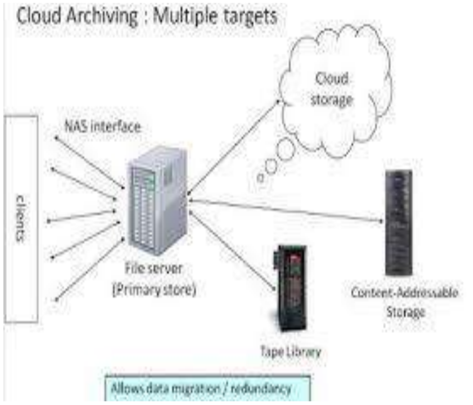
Cloud Archiving : Multiple targets
Cloud storage
NAS interface
clients
File server (Primary store)
Content-Addressable Storage
Tape Library
Allows data migration / redundancy

Fig. 3. Cloud storage

Present system

Here the data is stored in the third party cloud system, but it is a serious issue when concern to data confidentiality. So to provide strong confidentiality for data or messages in storage servers, user can encrypt messages by a cryptographic technique before applying an erasure code mechanism to encode and stored messages. When the user requires the message, the codeword symbols from storage servers has to be retrieved, decoded and then decrypted them by using cryptographic keys.

Drawbacks of present System

The problems that exist in the present encryption and encoding mechanism are:

Here the user has to do more computational task on its own,
The data traffic between the user and storage servers is very high, which may lead to congestion,
The user has to manage the cryptographic keys and security problems,
If the user's device that stores the keys is damaged or lost, then the security is broken,
It is very difficult for storage servers to directly support other operations or functions, besides handling data storage and retrieval functionalities.

Proposed system

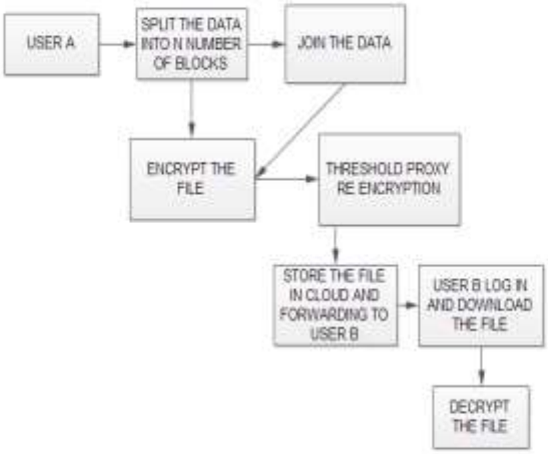
USER A
SPLIT THE DATA INTO N NUMBER OF BLOCKS
JOIN THE DATA
ENCRYPT THE FILE
THRESHOLD PROXY RE ENCRYPTION
STORE THE FILE IN CLOUD AND FORWARDING TO USER B
USER B LOG IN AND DOWNLOAD THE FILE
DECRYPT THE FILE

Fig 4: Overview architecture

The general architecture is shown in Fig. 4. In this architecture the user A splits the data into 'N' number of data block, combines all that data blocks into a file and then encrypts that using proxy re-encryption mechanism. After the file is encrypted it is stored or buffered in the cloud and forward to user B. User B then logs in and download the file and then decrypt the file.

In this paper we propose a proxy Re-encryption mechanism, here first the messages are encrypted by the owner and then stored in a storage server. When a user wants to share the messages, a re-encryption key is send to the storage server. Then the storage server re-encrypts the encrypted messages for the authorized user. Thus, this new scheme provides data confidentiality and also supports the data forwarding function.

Here the system model with distributed storage servers and key servers is considered.

The user distributes the cryptographic key to key servers that will perform cryptographic functions on behalf of the user, as it is risky to store the cryptographic key in a single device. These key servers consist of high security mechanisms to protect cryptographic keys [5].

To properly fit the distributed structure of systems, the servers should perform all operations independently. With this consideration, we propose a proxy re-encryption scheme and fuse it with a secure decentralized code to construct a secure distributed storage system. This proposed encryption design provides and strengthens encoding operations over encrypted messages and forwarding operations over encrypted and encoded messages. The tight fusion of encrypting, encoding and forwarding makes the storage system efficiently and effectively meet the prerequisites of data robustness, data confidentiality, and data forwarding.

Achieving the integration with consideration of a distributed structure is challenging. With this new mechanism the storage servers independently perform encoding and re-encryption functions, and the key servers perform partial decryption independently [6].

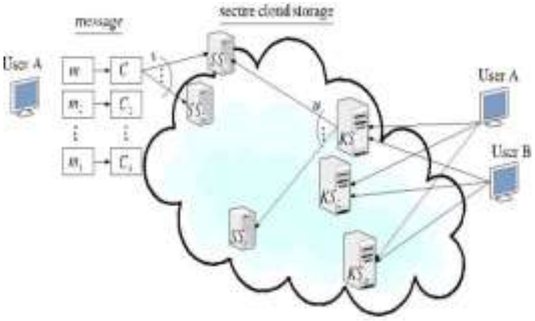

Fig.5. A general system model of the proposed work.

Scenario of cloud storage system

We represent the scenario of the storage system, the threat model that we consider for the confidentiality issue, and a discussion for a straightforward solution.

System prototype

Fig. 5 shows our system prototype of the proposed mechanism. This prototype consists of users, storage servers SS1, SS2, .. SSn, and m key servers KS1, KS2, .. KSm. Storage servers provide storage services and key servers provide key management services. They work independently. This model is a distributed storage system which consists of four procedural steps: system establishment, data storage, data forwarding, and data retrieval. These are discussed as follows:

In the system establishment phase, the system manager chooses system parameters and announces them. Each user A is allotted a public-secret key pair (PKA, SKA). User A distributes its secret key SKA to key servers so that each key server KSi holds a key share SKA,i, 1 ≤ i ≤ m.

In the data storage phase, user A encrypts its message M and dispatches it to storage servers. A message M is separated k blocks m1, m2,. . . , mk and each has an identifier tag. User A encrypts each block of message into a cipher text and sends it to randomly chosen storage servers. After receiving cipher text from the user, each storage server linearly combines them with randomly selected coefficients into a codeword symbol and stores it. The storage server may receive message blocks less than 'k' and can be assumed that all storage servers know the value k in advance [7].

In the data forwarding phase, user A forwards the encrypted message with an identifier ID stored in storage servers to user B so that B can decrypt the forwarded message by its secret key. To perform this decryption operation, A uses its secret key SKA and B's public key 'PKB' to calculate a re- encryption key *RKID A□□B* and sends *RKID A□□B* to all storage servers. Each storage server uses the same re-encryption key to re-encrypt its codeword symbol for later retrieval requests by B. The re-encrypted codeword symbol is the combination of cipher text under B's public key. In order to discriminate re-encrypted codeword symbols, we call them original codeword symbols and re-encrypted codeword symbols, respectively.

In the data retrieval phase, user A solicits for the retrieval of a message from storage servers, which is either stored by user A or forwarded to user A. User A sends a requisition to key servers to retrieve message. Upon receiving requisition and executing the authentication process with user A, each and every key server KSi requests randomly selected storage servers to acquire codeword symbols and will do partial decryption on the received codeword symbols by using the key share SKA,i. At last, user A associates the partially decrypted codeword symbols to extract the original message M. System recovering:

New storage sever is added in the network, when a storage server fails. The new storage server enquires ‘k’ available storage servers and linearly merges the received codeword symbols as a new code word and stores it. The system is then restored.

A secure cloud storage system with secure forwarding

The four procedural steps in the storage system:-

System Establishment: The algorithm Establishment (1τ) generates the system parameters μ. User A uses Key Generate(μ) to generate its public and secret key pair and Share Key Generate(k) to share its secret key to a set of ‘m’ key servers with a threshold value of ‘t’, where $k \leq t \leq m$. The third component of the user’s secret key is stored locally.

Data storage: When user A wants to buffer a message which of k blocks m1, m2, . . .,mk with an identifier ID, then the user A calculates the identity token τ =hf(a3,ID) and carry out the encryption algorithm Enc(.) on τ and ‘k’ blocks to get ‘k’ original cipher text C1, C2, . . . , Ck. The leading bit b = 0, indicates the original cipher text. User A transmits each cipher text Ci to ‘v’ casually selected storage servers. A set of original cipher text with the same identity token τ from user A is received by a storage server. When a cipher text Ci is not received, the storage server inserts Ci = (0, 1, τ, 1) to the set. The special format of (0, 1, τ, 1) is an indication mark for the absence of Ci. The storage server performs Encoding on the set of ‘k’ cipher text and stores the encoded result that is codeword symbol [8].

Data forwarding: If the User A wants to forward a message to another user B, it needs the first part a1 of its secret key. If A does not have a1, then it asks the key servers to share the key. When at least‘t’ key servers respond, user A recaptures the first part a1 of the secret key SKA with the help of the Key Recover algorithm. Here, let the identifier of the message be ‘ID’. User A calculates the encryption key *RKID A□□B* through the “Re Key Generate Algorithm” and sends the re-encryption key to each storage server safely and surely. By using *RKID A□□B,* a storage server re-encrypts the original codeword symbol ‘C’ with an identifier ID into a re-encrypted codeword symbol ‘C’ via the Re Encryption algorithm so that ‘C’ is decrypted by using user B’s secret key. A’s re-encrypted codeword symbol is specified by the leading bit b = 1. Let the public key PKB of user B be (gb1, hb2).

Data retrieval: Two situations for the data retrieval stage [9]. The first is that a user A extracts its own message. When user A requires the message with an identifier ‘ID’, it will notify all key servers with the identity token τ. A key server first gets original codeword symbols from randomly selected storage servers and then performs partial decryption Share Decrypt on every extracted original codeword symbol ‘C’. The result of partial decryption is called a partially decrypted codeword symbol. The key server transmits the partially decrypted codeword symbols ζ and the coefficients to user A. After user A gathers the acknowledgements from at least‘t’ key servers and at least ‘k’ of them are original from distinct storage servers, then the user A executes the ‘t’ partially decrypted codeword symbols to recapture the blocks m1, m2, . . ., mk. The second situation is that a user B extracts the message forwarded to it. User B notifies directly to all the key servers. Like the first situation, the collection and combining parts are the same, except that key servers extract the re-encrypted codeword symbols and perform partial decryption “Share Decrypt” on re-encrypted codeword symbols [10].

Conclusion

In this paper, a cloud storage system with storage servers and key servers is considered and new proposed threshold proxy re-encryption mechanism is integrated. This mechanism supports message encoding, message forwarding, and partial message decryption operations in a distributed manner. In order to decrypt a message of ‘k’ blocks which are encrypted and encoded to ‘n’ codeword symbols, each and every key server has to partially decrypt two codeword symbols in our proposed new system. By using the threshold proxy re-encryption mechanism, security is provided for cloud storage system which in turn provides secured data storage and secured data forwarding functionalities in a decentralized structure. In addition, each and every storage server independently accomplishes encoding and re-encryption process and each and every key server independently performs partial decryption. There is high compatibility and security in our storage system and with some other newly proposed content addressable file systems and storage system. In our proposed system, storage servers function as storage nodes in a content addressable storage system or like associative memory for storing content addressable blocks. Here the key servers enact as access nodes to provide a front-end layer such as a traditional file system interface. The mechanism discussed in this paper provides good security with authentication, message encoding and message forwarding.

References

1. J. Kubiatowicz, D. Bindel, Y. Chen, P. Eaton, D. Geels, R. Gummadi, S. Rhea, H. Weatherspoon, W. Weimer, C. Wells, and B. Zhao, “Oceanstore: An Architecture for Global-Scale Persistent Storage,” Proc. Ninth Int’l Conf. Architectural Support for Programming Languages and Operating Systems (ASPLOS), pp. 190- 201, 2000.
2. Software FEC in computer communications by Luigi Rizzo describes optimal erasure correction codes.
3. A. G. Dimakis, V. Prabhakaran and K. Ramchandran “Distributed Data Storage in Sensor Networks using Decentralized Erasure Codes”, Asilomar Conference on Signals, Systems and Computers, November 2004.
4. *McKinley PK, Samimi FA, Shapiro JK, Tang C.Service Clouds: A Distributed Infrastructure For Constructing Autonomic Communication Services. In Proceedings of the 2nd IEEE International Symposium on Dependable,Autonomic and Secure Computing,2006;341.*
5. S. Kamara and K. Lauter. Cryptographic cloud storage. In Financial Cryptography and Data Security (FC'10), volume 6054 of LNCS, pages 136{149. Springer, 2010.
6. M. v. Dijk, C. Gentry, S. Halevi, and V.Vaikuntanathan. Fully homomorphic encryption over the integers. In Advances in Cryptology { EUROCRYPT'10, volume 6110 of LNCS, pages 24{43. Springer, 2010.
7. C. Gentry and S. Halevi. Implementing Gentry's fully-homomorphic encryption scheme. Cryptology ePrint Archive, Report 2010/520, 2010. http://eprint.iacr.org/2010/520.
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. J. Algesheimer, C. Cachin, J. Camenisch, and G. Karjoth. Cryptographic security for mobile code. In Security and Privacy (S&P'01), pages 2{11. IEEE, 2001.
10. A. Shamir, “How to Share a Secret,” ACM Comm., vol. 22, pp. 612- 613, 1979.