*International Journal of Advanced Information Science and Technology (IJAIST) ISSN: 2319:268*
*Vol.2, No.5, May 2013 DOI:10.15693/ijaist/2013.v2i5.96-100*

# Effective Approach For Secure Storage Services In Cloud Computing

T .Deepika and M .Nisha
Department of Computer Science,
Vivekanandha College of Engineering and Technology For Women, Tiruchengode.

***ABSTRACT:*** *A cloud storage system consists of a collection of storage servers over the internet. The main aim is to provide secure storage services in a cloud storage system. Encryption schemes are used to provide data confidentiality, data robustness and functionality. Data confidentiality can be achieved through encryption but it limits the functionality of the storage system because a few operations are not supported over encrypted data. The storing cryptographic key in a single device is risky. It address the problem of forwarding data to another user by storage servers directly under the command of the data owner. Constructing a secure storage system that supports multiple functions is ever challenged for developers. The proposed system uses threshold proxy re-encryption scheme through decentralized erasure coding that provides secure distributed storage system. The distributed storage system not only supports secure and robust data storage and retrieval, but also a user forward his data in the storage servers to another user without retrieving the data back. The main technical contribution is that the proxy re-encryption scheme supports encoding operations over encrypted messages as well as forwarding operations over encoded and encrypted messages. It fully integrates encrypting, encoding, compressing and forwarding. It analyzes suitable parameters for the number of copies of a message dispatched to storage servers and the number of storage servers queried by a key server. And it supports dispersed storage reliability, scalability, reduced communication, and computation cost.*

***IndexTerms*—Secure storage system, Data security, Erasure coding, proxy re-encryption scheme.**

## I. INTRODUCTION

CLOUD computing is a new computing paradigm that is built on virtualization, parallel and distributed computing, utility computing, and service-oriented architecture. In the last several years, cloud computing has emerged as one of the most influential paradigms in the IT industry, Cloud computing is a concept that treats the resources on the Internet as a unified entity, a cloud. Users just use services without being concerned about how computation is done and storage is managed. It focuses on designing a cloud storage

system for robustness, confidentiality, and functionality. The cloud storage system is considered as a large scale distributed storage system that consists of many independent storage servers. Data robustness is a major requirement for storage systems. One way to provide data robustness is to replicate a message such that each storage server stores a copy of the message. It is very robust because the message can be retrieved as long as one storage server survives. Another way is to encode a message of k symbols into a codeword of n symbols by erasure coding. To store a message, each of its codeword symbols is stored in a different storage server. After the message symbols are sent to storage servers, each storage server independently computes a codeword symbol for the received message symbols and stores it. This finishes the encoding and storing process. The recovery process is the same.

The system model that consists of distributed storage servers and key servers. Since storing cryptographic keys in a single device is risky, a user distributes his cryptographic key to key servers that shall perform cryptographic functions on behalf of the user. The method of threshold proxy re-encryption scheme and integrate it with a secure decentralized code to form a secure distributed storage system. The encryption scheme supports encoding operations over encrypted messages and forwarding operations over encrypted and encoded messages. The tight integration of encoding, encryption, and forwarding makes the storage system efficiently meet the requirements of data robustness, data confidentiality, and data forwarding. The storage servers independently perform encoding and re-encryption and key servers independently perform partial decryption. The parameters are flexible adjustment between the number of storage servers and robustness.

## II. EXISTING SYSTEM

Storing data in a third party's cloud system causes serious concern on data confidentiality. To provide strong confidentiality for messages in storage servers, a user can encrypt messages by a cryptographic method before applying an erasure code method to encode and store messages. When he wants to use a message, he needs to retrieve the codeword symbols from storage servers, decode them, and then decrypt them by using cryptographic keys. There are three problems in the above straightforward integration of encryption and encoding. First, the user has to do most computation and the communication traffic between the user and storage servers is high. Second, the user has to manage his cryptographic keys. If the user's device of storing the keys is lost or compromised, the security is broken. Finally, data storing and retrieving, it is hard for storage servers to directly support other functions. For example, storage servers cannot directly forward a user's messages to another one. The owner of messages has to retrieve, decode, decrypt and then forward them to another user. It address the problem of forwarding data to another user by storage servers directly under the command of the data owner.

A decentralized architecture for storage systems offers good scalability, because a storage server can join or leave without control of a central authority. To provide robustness against server failures, a simple method is to make replicas of each message and store them in different servers. However, this method is expensive as z replicas result in z times of expansion. One way to reduce the expansion rate is to use erasure codes to encode messages. A message is encoded as a codeword, which is a vector of symbols, and each storage server stores a codeword symbol. A storage server failure is modeled as an erasure error of the stored codeword symbol. Random linear codes support distributed encoding, that is, each codeword symbol is independently computed. To store a message of k blocks, each storage server linearly combines the blocks with randomly chosen coefficients and stores the codeword symbol and coefficients. To retrieve the message, a user queries k storage servers for the stored codeword symbols and coefficients and solves the linear system. It considered the case that n= ak for a fixed constant a. A decentralized erasure code is an erasure code that independently computes each codeword symbol for a message. Thus, the encoding process for a message can be split into n parallel tasks of generating codeword symbols. A decentralized erasure code is suitable for use in a distributed storage system. After the message symbols are sent to storage servers, each storage server independently computes a codeword symbol for the received message symbols and stores it. A message is encoded as a codeword, which is a vector of symbols, and each storage server stores a codeword symbol. A storage server failure is modelled as an erasure error of the stored codeword symbol. Random linear codes support distributed encoding, that is, each codeword symbol is independently computed. To store a message of k blocks, each storage server linearly combines the blocks with randomly chosen coefficients and stores the codeword symbol and coefficients. To retrieve the message, a user queries k storage servers for the stored codeword symbols and coefficients and solves the linear system.

The sparsity parameter v = b ln k is the number of storage servers which a block is sent to. The larger v is, the communication cost is higher and the successful retrieval probability is higher. The system has light data confidentiality because an attacker can compromise k storage servers to get the message. In a proxy re-encryption scheme, a proxy server can transfer a ciphertext under a public key PKA to a new one under another public key PKB by using the re-encryption key RKA-B. The server does not know the plaintext during transformation. The some proxy re-encryption schemes and applied them to the sharing function of secure storage systems.

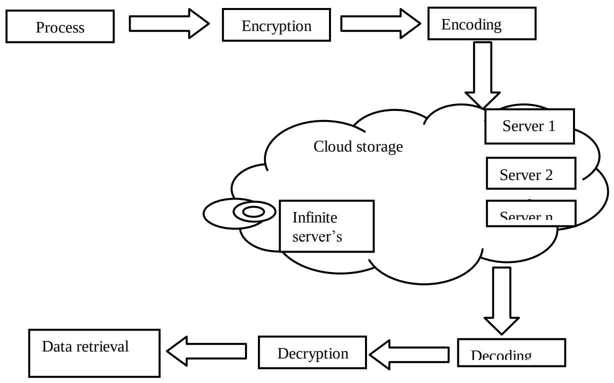


**Fig 1: Architecture for existing system**:

As in show fig1, the first encrypted and encoding by the owner and then stored in a storage server. When a user wants to share his messages, he sends a re-encryption key to the storage server. The storage server re-encrypts the encrypted messages for the authorized user. Thus, their system has data confidentiality and supports the data forwarding function. It further integrates encryption, re-encryption, and encoding such that storage robustness is strengthened. Our system model consists of users, n storage servers SS1; SS2;.. SSn. A user gets an original message from a data retrieval process.

A straightforward solution to supporting the data forwarding function in a distributed storage system Proxy re-encryption schemes can significantly decrease communication and computation cost of the owner. In a proxy re-encryption scheme, the owner sends a re-encryption key to storage servers such that storage servers perform the re-encryption operation for him. Thus, the communication cost of the owner is independent of the length of forwarded message and the computation cost of re-encryption is taken care of by storage servers. Proxy re-encryption schemes significantly reduce the overhead of the data forwarding function in a secure storage system. It analyzes storage and computation complexities, correctness, and security of our cloud storage system. It will reduce the storage cost.

## III.PROPOSED SYSTEM

The system model consists of two distributed storage servers and servers. There are three methods integrate to forming a secure cloud storage system and it supports a function of secure data forwarding, they are

1) Decentralised erasure code
2) Threshold proxy re-encryption scheme
3) Data optimization.

### A. Decentralised Erasure Code

A decentralized erasure code is an erasure code that independently computes each codeword symbol for a message.

Thus, the encoding process for a message can be split into n parallel tasks of generating codeword symbols. A decentralized erasure code is suitable for use in a distributed storage system. After the message symbols are sent to storage servers, each storage server independently computes a codeword symbol for the received message symbols and stores it. A message is encoded as a codeword, which is a vector of symbols, and each storage server stores a codeword symbol. A storage server failure is modelled as an erasure error of the stored codeword symbol. Random linear codes support distributed encoding, that is, each codeword symbol is independently computed. To store a message of k blocks, each storage server linearly combines the blocks with randomly chosen coefficients and stores the codeword symbol and coefficients. To retrieve the message, a user queries k storage servers for the stored codeword symbols and coefficients and solves the linear system.

### *A.1.Randomized Network Algorithm*

There is a very simple, robust randomized algorithm to construct a decentralized erasure code in a network. Each data node picks one out of the n storage nodes randomly, pre-routes its packet and repeats d(k) = c ln(k) times. Each storage node multiplies whatever it happens to receive with coefficients selected uniformly and independently in $F_q$ and stores the result and the coefficients. A schematic representation of this is given in Figure 2.

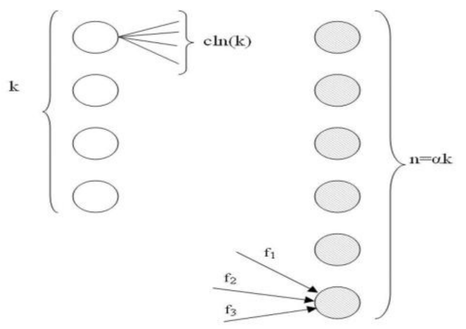


Fig2. Decentralized erasure codes construction

The key advantage of decentralized erasure codes is that *there is no need for coordination* among the data nodes. We show how data nodes acting randomly and independently, can create good erasure codes with sparse structure. We have completely characterized how sparse the generator matrices of these codes can be, and present a simple randomized construction algorithm that requires no centralized coordination. The property of decentralized erasure codes makes them ideal for scenarios where data is distributed and global coordination is difficult. The generator matrix of the decentralized erasure codes. I never constructed explicitly and does not even exist in one place in the network.

### *B. Proxy Re-Encryption Scheme*

In a proxy re-encryption scheme, a proxy server can transfer a ciphertext under a public key PKA to a new one under another public key PKB by using the re-encryption key RKA-B. The server does not know the plaintext during transformation. The some proxy re-encryption schemes and applied them to the sharing function of secure storage systems. It use threshold proxy re-encryption scheme with multiplicative homomorphic property. An encryption scheme is multiplicative homomorphic property. And it supports the encoding operation over encrypted messages. Then it converts proxy re-encryption scheme with multiplicative homomorphic property into a threshold version.

The messages are first encrypted by the owner and then stored in a storage server. When a user wants to share his messages, he sends a re-encryption key to the storage server. The storage server re-encrypts the encrypted messages for the authorized user. Thus, their system has data confidentiality and supports the data forwarding function. Our work further integrates encryption, re-encryption, and encoding such that storage robustness is strengthened.

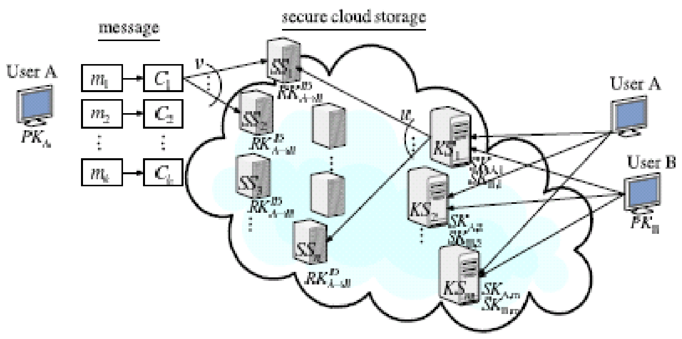


Fig 3: System Model

As in show Fig3, the our system model consists of users, n 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. Our distributed storage system consists of four phases: system setup, data storage, data forwarding, and data retrieval. The four phases are described as follows.

In the system setup phase, the system manager chooses system parameters and publishes them. Each user A is assigned a public-secret key pair (PKA, SKA). User A distributes his secret key SKA to key servers such that each key server KSi holds a key share SKA;i , 1 < i < m. The key is shared with a threshold t.

In the 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 a ciphertext Ci and sends it to v randomly chosen storage servers. Upon receiving ciphertext from a user, each storage server linearly combines them with randomly chosen coefficients into a codeword symbol and stores it. The storage server may receive less than k message blocks and it assume that all storage servers know the value k in advance.

In the data forwarding phase, user A forwards his encrypted message with an identifier ID stored in storage servers to user B such that B can decrypt the forwarded message by his secret key. To do so, A uses his secret key SKA and B's public key PKB to compute a re-encryption key $RK^{ID}_{A\to B}$ and then sends $RK^{ID}_{A\to B}$ to all storage servers. Each storage server uses the re-encryption key to re-encrypt its codeword symbol for later retrieval requests by B. The re-encrypted codeword symbol is the combination of ciphertext under B's public key. In order to distinguish re-encrypted codeword symbols from intact ones, we call them original codeword symbols and re-encrypted codeword symbols, respectively.

In the data retrieval phase, user A requests to retrieve a message from storage servers. The message is either stored by him or forwarded to him. User A sends a retrieval request to key servers. Upon receiving the retrieval request and executing a proper authentication process with user A, each key server KSi requests u randomly chosen storage servers to get codeword symbols and does partial decryption on the received codeword symbols by using the key share SKA;i. Finally, user A combines the partially decrypted codeword symbols to obtain the original message M.

### *C. Data Optimization Techniques*

The optimization techniques are deduplication and compression, which are provided by some services in order to save bandwidth.

#### *C.1 Deduplication*

The deduplication process is a popular technique that allows cloud storage providers to significantly decrease the amount of needed storage space. The principle of deduplication is as follows: only a single copy of each piece of data is stored. If a user wants to store data that the cloud storage provider already has stored in the past, the storage provider simply creates a link to that data instead of storing another copy. The File level deduplication means that only a single copy of each file will be stored. Block level deduplication means that each block will be split up into blocks and only a single copy of each block will be stored. Identical files or blocks are detected by comparing the hash value" with a list of known files or blocks.

#### *C.2 Compression*

A simple way to save bandwidth is to compress data on client-side. To encrypt the data's SHA Hash Algorithm is used for compression GZIP algorithm.

### *D. A Secure Cloud Storage System with Secure Forwarding*

*System setup*: The algorithm SetUp ( ) generates the system parameters μ. A user uses KeyGen (μ) to generate his public and secret key pair and ShareKeyGen () to share his secret key to a set of m key servers with a threshold t, where $k \le t \le m$. The user locally stores the third component of his secret key.
*Data storage*: When user A wants to store a message of k blocks m1;m2; . . .;mk with the identifier ID, he computes the identity token τ and performs the encryption algorithm Enc(PKA; τ;m1;m2; . . .;mk ) and Encode(C1,C2,….Ck ) and k blocks to get k original ciphertexts C1; C2; . . . ; Ck.
*Data forwarding:* 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, he queries key servers for key shares. When at least t key servers respond, A recovers the first component a1 of the secret key SKA via the KeyRecover(SKA;i1 ; SKA;i2 ; . . . ; SKA; it) algorithm. Let the identifier of the message be ID. User A computes the re-encryption key $RK^{ID}_{A\to B}$ via the ReKeyGen (PKA; SKA; ID;PKB ) algorithm and securely sends the re-encryption key to each storage server. By using RKIDA!B, a storage server re-encrypts the original codeword symbol C' with the identifier ID into a re-encrypted codeword symbol C'' via the ReEnc($RK^{ID}_{A\to B}$ , C'') algorithm such that C'' is decryptable by using B's secret key.
*Data retrieval*: There are two cases for the data retrieval phase. The first case is that a user A retrieves his own message. When user A wants to retrieve the message with the identifier ID, he informs all key servers with the identity token T. A key server first retrieves original codeword symbols from u randomly chosen storage servers and then performs partial decryption ShareDec() on every retrieved original codeword symbolC0. The result of partial decryption is called a partially decrypted codeword symbol. The key server sends the partially decrypted codeword symbols and the coefficients to user A. After user A collects replies from atleast t key servers and at least k of them are originally from distinct storage servers, he executes Combine() on the t partially decrypted codeword symbols to recover the blocks m1;m2; . . .;mk. The second case is that a user B retrieves a message forwarded to him. User B informs all key servers directly. The collection and combining parts are the same as the first case except that key servers retrieve re-encrypted codeword symbols and perform partial decryption ShareDec() on re-encrypted codeword symbols.

## E.ANALYSIS

It analyzes the storage and computation complexities, correctness, and security of our cloud storage system.

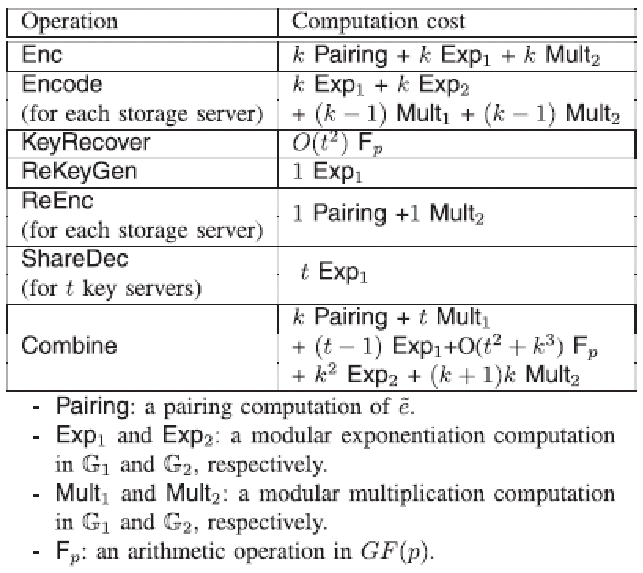
*Storage cost:*

To store a message of k blocks, a storage server stores a codeword symbol and the coefficient vector.It will reduce the storage cost in each storage server.

Computation cost:

It measure the computation cost by the number of pairing operations, modular exponentiations in GG1 and GG2, modular multiplications in GG1 and GG2, and arithmetic operations over GF(p).

TABLE 1
The Computation Cost of Each Algorithm
in Our Secure Cloud Storage System



| Operation | Computation cost |
|---|---|
| Enc | $k$ Pairing + $k$ Exp$_1$ + $k$ Mult$_2$ |
| Encode (for each storage server) | $k$ Exp$_1$ + $k$ Exp$_2$ + $(k-1)$ Mult$_1$ + $(k-1)$ Mult$_2$ |
| KeyRecover | $O(t^2)$ F$_p$ |
| ReKeyGen | 1 Exp$_1$ |
| ReEnc (for each storage server) | 1 Pairing +1 Mult$_2$ |
| ShareDec (for $t$ key servers) | $t$ Exp$_1$ |
| Combine | $k$ Pairing + $t$ Mult$_1$ + $(t-1)$ Exp$_1$+O$(t^2+k^3)$ F$_p$ + $k^2$ Exp$_2$ + $(k+1)k$ Mult$_2$ |

- Pairing: a pairing computation of $\tilde{e}$.
- Exp$_1$ and Exp$_2$: a modular exponentiation computation in $\mathbb{G}_1$ and $\mathbb{G}_2$, respectively.
- Mult$_1$ and Mult$_2$: a modular multiplication computation in $\mathbb{G}_1$ and $\mathbb{G}_2$, respectively.
- F$_p$: an arithmetic operation in $GF(p)$.

## *IV CONCULSION AND FUTURE WORK*

The cloud storage system consists of storage servers and key servers. The method of threshold proxy re-encryption scheme, data optimization technique and erasure codes over exponents. The proxy re-encryption scheme supports encoding, forwarding, and partial decryption operations in a distributed way. To decrypt a message of k blocks that are encrypted and encoded to n codeword symbols, each key server only has to partially decrypt two codeword symbols in our system. By using the threshold proxy re-encryption scheme, it present a secure cloud storage system that provides secure data storage and secure data forwarding functionality in a decentralized structure. Each storage server independently performs encoding and re-encryption and each key server independently perform partial decryption. It will reduce the storage and computation cost.

The storage system and some newly proposed content addressable file systems and storage system are highly compatible. Our storage servers act as storage nodes in a content addressable storage system for storing content addressable blocks. Our key servers act as access nodes for providing a front-end layer such as a traditional file system interface.

## REFERENCES

1. 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.
2. 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.
3. 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.
4. Ungureanu .C, Atkin .B, Aranya .A, Gokhale .S, Rago .S, Calkowski .G,Dubnicki .C, and Bohra .A,(2010) “Hydrafs: a high-throughput file system for the hydrastor content addressable storage system,” in Proceedings of the 8th USENIX Conference on File and Storage Technologies - FAST, p. 17,USENIX.
5. C. Dubnicki, L. Gryz, L. Heldt, M. Kaczmarczyk, W. Kilian, P. Strzelczak, J. Szczepkowski, C. Ungureanu, and M. Welnicki, “Hydrastor: A Scalable Secondary Storage,” Proc. Seventh Conf. File and Storage Technologies (FAST), pp. 197-210, 2009.
6. W. Dong, F. Douglis, K. Li, H. Patterson, S. Reddy, and P. Shilane, “Tradeoffs in Scalable Data Routing for Deduplication Clusters,”Proc. Ninth USENIX Conf. File and Storage Technologies (FAST), p. 2,
2011.
7. G. Ateniese, R.D. Pietro, L.V. Mancini, and G. Tsudik, “Scalable and Efficient Provable Data Possession,” Proc. Fourth Int’l Conf. Security and Privacy in Comm. Netowrks (SecureComm), pp. 1-10, 2008.
8. 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.
9. Haeberlen .A, Mislove .A, and Druschel .P, “Glacier: Highly durable,decentralized Storage despite massive correlated failures,” in Proceedingsofthe 2nd Symposium onNetworked Systems Design and Implementation.
10. S. Yu, C. Wang, K. Ren, and W. Lou, “Achiving secure, scalable, and fine-grained data access control in cloud computing,” in Proc. IEEE INFOCOM 2010, 2010, pp. 534–542