International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 06 Issue: 3 | Mar 2019 www.irjet.net p-ISSN: 2395-0072

A SECURE ERASURE CODE-BASED CLOUD STORAGE FRAMEWORK WITH SECURE INFORMATION SENDING

Pavithra. K[1] and Prakash Narayanan. C[2]
paviselvi121@gmail.com and cprakashmca@gmail.com
[1]PG Student and [2]Assistant Professor, Department of Computer Science and Engineering
P.S.V. College of Engineering & Technology, Krishnagiri.

Abstract - A distributed storage framework, comprising of an accumulation of capacity servers, gives long haul stockpiling administrations over the Internet. Putting away information in an outsider's cloud framework causes genuine worry over information secrecy. General encryption plans ensure information privacy, yet in addition limit the usefulness of the capacity framework in light of the fact that a couple of activities are upheld over encoded information. Building a safe stockpiling framework that underpins different capacities is testing when the capacity framework is disseminated and has no focal expert. We propose an edge intermediary re-encryption conspire and incorporate it with a decentralized eradication code to such an extent that a safe appropriated stockpiling framework is detailed. The circulated stockpiling framework not just backings secure and strong information stockpiling and recovery, yet additionally lets a client forward his information in the capacity servers to another client without recovering the information back. The primary specialized commitment is that the intermediary re-encryption plot bolsters encoding tasks over scrambled messages just as sending activities over encoded and scrambled messages. Our strategy completely incorporates scrambling, encoding, and sending. We examine and propose reasonable parameters for the quantity of duplicates of a message dispatched to capacity servers and the quantity of capacity servers questioned by a key server. These parameters permit increasingly adaptable alteration between the quantity of capacity servers and power.

Key Words: *Threshold cryptography, secure storage system, proxy re-encryption*

1. INTRODUCTION

As high-speed networks and ubiquitous Internet access become available in recent years, many services are provided on the Internet such that users can use them from anywhere at any time. For example, the email service is probably the most popular one. 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. In this paper, we focus on designing a cloud storage system for robustness, confidentiality, and functionality.

A 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. There have been many proposals of storing data over storage servers . 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.

2. PROPOSED SYSTEM

- We construct a secure cloud storage system that supports the function of secure data forwarding by using a threshold proxy re-encryption scheme. The encryption scheme supports decentralized erasure codes over encrypted messages and forwarding operations over encrypted and encoded messages. Our system is highly distributed where storage servers independently encode and forward messages and key servers independently perform partial decryption.
- We present a general setting for the parameters of our secure cloud storage system. Our parameter setting of supersedes the previous one. Our result allows the number of storage servers be much greater than the number of blocks of a message. In practical systems, the number of storage servers is much more than k. The sacrifice is to slightly increase the total copies of an encrypted message symbol sent to storage servers. Nevertheless, the storage size in each storage server does not increase because each storage server stores an encoded result (a codeword symbol), which is a combination of encrypted message symbols

3. EXISTING SYSTEM

- Storing data in a third party's cloud system causes serious concern on data confidentiality. In order 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, besides 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.

4. A STRAIGHTFORWARD SOLUTION

A straightforward solution to supporting the data forwarding function in a distributed storage system is as follows: when the owner A wants to forward a message to user B, he downloads the encrypted message and decrypts it by using his secret key. He then encrypts the message by using B's public key and uploads the new ciphertext. When B wants to retrieve the forwarded message from A, he downloads the ciphertext and decrypts it by his secret key. The whole data forwarding process needs three communication rounds for A's downloading and uploading and B's downloading.

The communication cost is linear in the length of the forwarded message. The computation cost is the decryption and encryption for the owner A, and the decryption for user B. Proxy re-encryption schemes can significantly decrease communication and computation cost of the owner. 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.

5. A SECURE CLOUD STORAGE SYSTEM WITH SECURE FORWARDING

There are four phases of our storage system. System setup. The algorithm Setup generates the system parameters. 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 t m. The user locally stores the third component of his secret key.

5.1 Analysis

We analyze storage and computation complexities, correctness, and security of our cloud storage system in this section. Let the bit-length of an element in the group G1 be l1 and G2 be l Let coefficients be randomly chosen from . Storage cost. To store a message of k blocks, a storage server SSj stores a codeword symbol and the coefficient vector . They are total of bits, where j; 2 G1 and j 2 G2.

The average cost for a message bit stored in a storage server is bits, which is dominated by l3=l2 for a sufficiently large k. In practice, small coefficients, , reduce the storage cost in each storage server. Computation cost. We measure the computation cost by the number of pairing operations, modular exponentiations in G1 and G2, modular multiplications in G1 and G2, and arithmetic operations over GF. These operations are denoted as Pairing, Exp1, Exp2, Mult1, Mult2, and Fp, respectively.

5.2 Scenario

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

5.3 System Model

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. These 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 publicsecret key pair (PKA; SKA).

5.4 Threat Model

We consider data confidentiality for both data storage and data forwarding. In this threat model, an attacker wants to break data confidentiality of a target user. To do so, the attacker colludes with all storage servers, non target users, and up to 1Þ key servers. The attacker analyzes stored messages in storage servers, the secret keys of non target users, and the shared keys stored in key servers.

Note that the storage servers store all re-encryption keys provided by users. The attacker may try to generate a new re-encryption key from stored re-encryption keys. We formally model this attack by the standard chosen plaintext attack1 of the proxy 1. Systems against chosen ciphertext attacks are more secure than systems

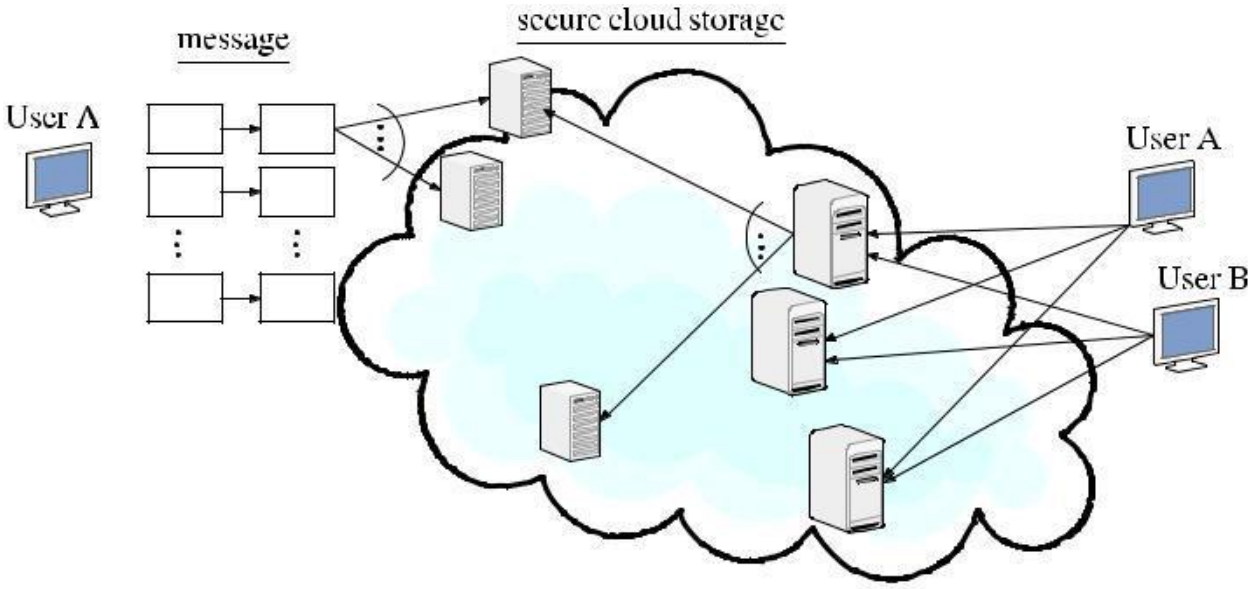

Fig.5.1 Threat Model

6. INTEGRITY CHECKING FUNCTIONALITY

Another important functionality about cloud storage is the function of integrity checking. After a user stores data into the storage system, he no longer possesses the data at hand. The user may want to check whether the data are properly stored in storage servers. The concept of provable data possession [20], [21] and the notion of proof of storage [22], [23], [24] are proposed. Later, public auditability of stored data is addressed in [25]. Nevertheless all of them consider the messages in the cleartext form 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 ciphertexts from a user, each storage server linearly combines them with randomly chosen coefficients into a codeword symbol and stores it. Note that 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, 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 RKIDA!B and then sends RKIDA!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 ciphertexts under B's public key. In order to distinguish reencrypted codeword symbols from intact ones, we call them original codeword symbols and re-encrypted codeword

7. CONCLUSION

In this paper, we consider a cloud storage system consists of storage servers and key servers. The proposed threshold proxy re-encryption scheme and erasure codes over exponents. The threshold 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, we present a secure cloud storage system that provides secure data storage and secure data forwarding functionality in a decentralized structure. Moreover, each storage server independently performs encoding and re-encryption and

each key server independently performs partial decryption. Our 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. 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 System Design and Implementation (OSDI), pp. 1-14, 2002.

[2] A. Haeberlen, A. Mislove, and P. Druschel, "Glacier: Highly Durable, Decentralized Storage Despite Massive Correlated Fail-ures," Proc. Second Symp. Networked Systems Design and Implemen-tation (NSDI), pp. 143-158, 2005.

[3] 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.

[4] 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.

[5] 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.

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

[7] 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.