RESEARCH ARTICLE OPEN ACCESS

FMSR-AES in Multiple Cloud Storage System to Provide Secured Fault Tolerant

V.Swathy [1], T.S Usha Nandhini [2], T.K.P Rajagopal [3]
PG Scholar [1], Assistant Professor/Guide [2], Assistant Professor [3]
Department of Computer Science and Engineering
Kathir College of Engineering
Neelambur,Coimbatore
TamilNadu –India

ABSTRACT
Cloud computing is the delivery of services on-demand through internet. Storing data in the hard drives are less reliable and create space constraints. To make the data available online and access anywhere without space constraints, cloud storage is evolved. Though cloud storage has many advantages over traditional storage methods, it also have a notable problem to be addressed which is fault tolerance. Regenerating codes stripes data to multiple servers and makes the storage fault tolerant during failures. Protection of data against corruptions along with fault tolerance becomes more challenging. To address that, we propose Advance Encryption Standards (AES) on the top of Frequency Minimum-Storage Regenerating (FMSR) as solution for the above problems. Our work proves the efficiency in computation time for adding security to the data at rest when compared to other methods such as FMSR.
Keywords: - Advance Encryption Standards, Cloud storage, Regenerating codes.

I. INTRODUCTION

With the Cloud computing, the computing architecture have changed drastically and so the delivery models. Cloud delivers everything as service, so that the organizations need not to do initial investments. The delivery model and service models differs with the user's requirement. Outsourcing data to a third party to store and make it ubiquitous is Storage-as-A-Service. To reduce the Used on the top of FMSR codes. Retrievability proof [11] and data possession proof [3] proposed to test the originality of a huge file by checking the parts of the file using different methods of cryptography. When we outsource our data to a storage service provider and we observe modification in our stored data, then that data must be regenerated and reconstruct the original file. Storing the data in a cloud may be prone to the failure problem [2] and the works [1], [2], suggested to split and store data in more than one storages. Thus, to restore a storage data followed by a stable failure, the steps followed are

1) Read data from the other live servers,
2) Regenerate the corrupted data of the server met with failure, and
3) Regenerated data on a new server. Retrievability

Proof [11] and Data Possession Proof [3] are originally have been developed for single-servers. In cost of cloud storage and maintenance like power or other risks, the cloud storage provides decided to move the data to trusted third party storage providers. This makes the provider's job easier. As the third party storages increases, the security and also the integrity of the stored data are put into risk. When cloud faces failure, users should not be suffered from data loss which reduces the QoS. To provide fault tolerance and integrity verification against failure due to corruption, the encryption method AES-128.

[7] and [4] the data originality for multi-server have been observed. Regenerating codes [8] introduced to reduce the movement of data while restoring it. Additionally, it is not done by regenerating the entire file as in erasure coding, however taking a chunk of a file from other live storages and it's been regenerated the chunk which is missed due to failures was observed in NCCloud [15]. Thus the fault tolerant is provided to the data stored in the multiple cloud storage.

II. RELATED WORK

We closely analyzed the previous works and the security of stored data in the third party cloud storages. Only a single server problems are considered by Juels and Kaliski [11] created POR and Ateniese et al. [3] created PDP. A major problem of the above schemes are they are consideration of single storage servers. If the storage is attacked some intruder, the system can only capable of detecting of

failures but not the regeneration. Then the schemes in a multiserver setting for checking data for its originality. By splitting and storing data in more than one storage can be able to regenerate the data from the peer live storages. Since, replication [7] and erasure coding [4], [5] provided data to be stored in multiple sites may encourage hacking of valuable data. Though, Chen et al. [6] proposed storage by using regenerating codes, our work differs with the security aspect.

The storage methods proposed [6] shows that storage servers should adapt the encoding techniques for generating data, still for a thin-cloud [13] have been developed, whose storage have the capacity of read and write operations. Our work with linked with HAIL but it is operated based on single file at a time. Patrick in his work FMSR-DIP [9] provided security for the data though single threaded mechanism. Though the security is good but the time taken for encryption is high. William A.Walls [7] proposed integrity check for data alone. All the above works are based on the cloud storage security. However, those schemes which cannot be adapted for the FMSR codes on the proxy server system.

III. PROBLEM DESCRIPTION

The problems persist in the existing Proxy system based FMSR coding security of the data being outsourced for storage has not been ensured.

A.Cloud Security Issues

When data at rest on the cloud storage, there may be hackers who can try to hack and steal data. This makes the user to suffer from altered

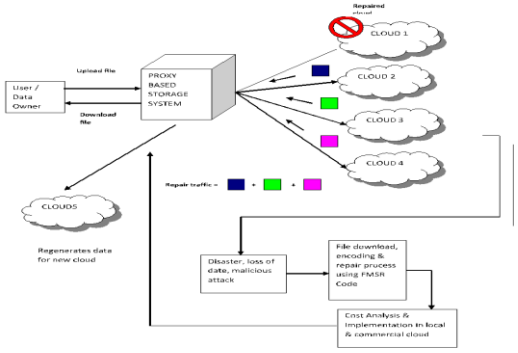


Fig 1. Architecture FMSR codes

data even it is regenerated from the failures. With the stolen data, hacker can misuse it. Even the data is of no use for the user when it is placed in the new storage by the regenerating codes. To avoid this problem, the data must be encrypted in the proxy server after it is chunked and encoded by the FMSR.

IV. OUR CONTRIBUTION

In our proposed scheme, to provide security for the data at rest in the different storage the data is to be encrypted and the corresponding the FMSR coding can be done on its top to chunk and encode the data. The proxy server in the system must be responsible for providing the encryption to the data being stored. Firstly, after receiving data from the user, the proxy server must encrypt the data using AES – 126 algorithm. The reason for choosing AES is it provides resistance to all known attacks, speed and simplicity in design and coding.

A.AES – 126 Basic Operations

The encryption operation starts with add round key stage then, nine rounds of four stages and a tenth round of three stages followed by the add round key stage. Stages mentioned above are

1) SubBytes

It is nothing but the byte values forms a lookup table of 16×16 matrix. Matrix contains all the permutation of sequence of 8 bits (28= 16 × 16 = 256).

2) Shift Row Transformation

This stage is known as ShiftRows and it performs a simple permutation. The steps performed are

- The first row of state doesn't get altered.
- The second row is left shifted by 1 byte.
- The third row is left shifted by 2 bytes.
- The fourth row is left shifted by 3 bytes.

3) Mix Column Transformation

This is basically a substitution operation. Each column is operated add round key. It takes the 4 bytes of a state and one word of the round key in the matrix and the operations are done. This operation affects all the bits in the file.

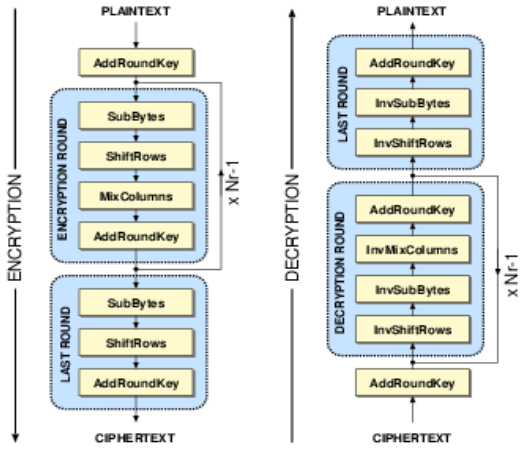


Fig 2. Basic Operations of AES-128

The decryption operation involves the inverse of all the operations mentioned above in reverse order. The last round leaves the mix column state individually. A new bit is generated with the mapping of every column value and the four bytes in the matrix.

4) Transformation of Add Round Key

In this stage, every bit of state is XOR with the 128 bits of the data.

V. FMSR-AES IMPLEMENTATION

The operations involved in FMSR codes are File distribution from proxy, File regeneration and Repair operation. Here we have added steps chunk encryption and chunk decryption in the first and last operations respectively. During *File distribution from proxy*, FMSR-AES codes, first chunk the file with FMSR encoding technique described in [1]. After the chunking and encoding is done, AES – 128 is being applied on top of that. This operation ensures that the third party server which stores the data chunk cannot get the raw data as the secret key will not be shared with them. Thus the whole encryption process is taken care by the proxy server in FMSR. During *File Regeneration*, the when failure is detected by the proxy server, it must regenerate the file from other live servers which stores the metadata of other failed servers. It uses the same process followed in FMSR with the added decryption of AES from the secret key kept in the proxy server. The data which is obtained after the decryption process has the data chunk encoded with other chunks. During *Repair* operation, the data chunks are regenerated and being written on the new cloud storage.

A. FMSR – AES algorithm

//Pseudo code for File distribution from proxy
i. Get data file from user
ii. Chunk the file into equal sizes with the number same as the multiple storage
iii. Encode the data with the data from other chunks
iv. Encrypt each chunk with AES-128 algorithm with the secret key
v. Distribute the chunks to each of store storage

//Pseudo code for File regeneration
i. After sensing the failure, data must be retrieved back to the proxy
ii. Regenerate by decoding and decrypting with the same algorithm

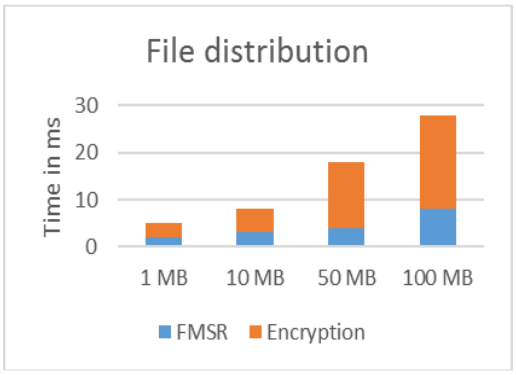
//Pseudo code for repair
i. Data is reconstructed to a new cloud storage
ii. After user request, data must be retrieved back to the user

VI. RESULT ANALYSIS

The FMSR-AES is implemented in the local cloud environment and an artificial failure is induced. Thus the reconstruction of data with the encryption and decryption is observed.
The failures induced are

1) Permanent by deleting a file
2) doze-off
3) Hacked data

The below graphs show the computational time which is captured during each operations in our FMSR-AES process.

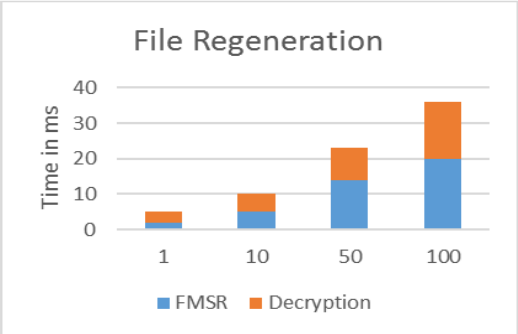


a) File distribution from proxy

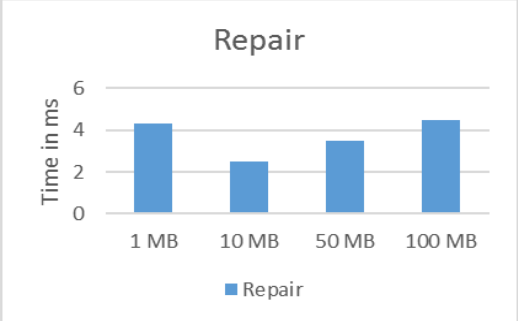
b)

Graph (a) shows the time taken for both encryption and FMSR encoding process. It is observed that the encryption process takes more time, since it adds more security any cloud provider can afford it. Graph (b) and (c) shows that the time taken

for regeneration which involved both the decoding and decryption using the secret key secured by the proxy server and repair process respectively.



b) File Regeneration



c) Repair

VII. CONCLUSION

The popularity of outsourced long term archival data storage is increasing, it is essential for the user to make sure that their data is secured. We designed and implemented AES-128 algorithm for FMSR codes. Thus FMSR-AES codes preserves the fault tolerance with the security to the data stored in multiple storage servers.

REFERENCES

[1] H. Abu-Libdeh, L. Princehouse, and H. Weatherspoon, "RACS: A Case for Cloud Storage Diversity," Proc. First ACM Symp. Cloud Computing (SoCC '10), 2010.

[2] M. Armbrust, A. Fox, R. Griffith, A.D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, and M.Zaharia, "A View of Cloud Computing," Comm. ACM, vol. 53, no. 4, pp 50-58, 2010.

[3] G. Ateniese, R. Burns, R. Curtmola, J. Herring, O. Khan, L.Kissner, Z. Peterson, and D. Song, "Remote Data Checking Using Provable Data Possession," ACM Trans. Information and System Security, vol. 14, article 12, May 2011.

[4] K. Bowers, A. Juels, and A. Oprea, "HAIL: A High-Availability and Integrity Layer for Cloud Storage," Proc. 16th ACM Conf. Computer and Comm. Security (CCS '09), 2009.

[5] K. Bowers, A. Juels, and A. Oprea, "Proofs of Retrievability: Theory and Implementation," Proc. ACM Workshop Cloud Computing Security (CCSW '09), 2009.

[6] B. Chen, R. Curtmola, G. Ateniese, and R. Burns, "Remote Data Checking for Network Coding-Based Distributed Storage Systems," Proc. ACM Workshop Cloud Computing Security (CCSW '10), 2010.

[7] R. Curtmola, O. Khan, R. Burns, and G. Ateniese, "MR-PDP: Multiple- Replica Provable Data Possession," Proc. IEEE 28th Int'l Conf. Distributed Computing Systems (ICDCS '08), 2008.

[8] A. Dimakis, P. Godfrey, Y. Wu, M. Wainwright, and K. Ramchandran, "Network Coding for Distributed Storage Systems,"IEEE Trans. Information Theory, vol. 56, no. 9, 4539-4551,Sept. 2010.

[9] Henry C.H. Chen and Patrick P.C. Lee, "Enabling Data Integrity Protection in Regenerating-Coding-Based Cloud Storage: Theory and Implementation," IEEE Trans. Parallel and distributed systems, vol. 25, no. 2, 407,416, Feb 2014.

[10] Y. Hu, H. Chen, P. Lee, and Y. Tang, "NCCloud: Applying Network Coding for the Storage Repair in a Cloud-of- Clouds," Proc. 10th USENIX Conf. File and Storage Technologies (FAST '12), 2012.

[11] A. Juels and B. Kaliski Jr., "PORs: Proofs of Retrievability for Large Files," Proc. 14th ACM Conf. Computer and Comm. Security (CCS '07), 2007.

[12] "TechCrunch," Online Backup Company Carbonite Loses Customers' Data, Blames and SuesSuppliers, http://techcrunch.com/2009/03/23/online-backup-company-carbonite-losescustomers-datablames-and suessuppliers/, Mar. 2009.

[13] M. Vrable, S. Savage, and G. Voelker, "Cumulus: Filesystem Backup to the Cloud," Proc. USENIX Conf. File and Storage Technologies (FAST), 2009.