ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

*(An ISO 3297: 2007 Certified Organization)*

**Vol. 3, Issue 12, December 2015**

# Data Integrity Protection and Key Aggregation for Data Sharing In Cloud Computing

**B. Sivasakthi, V. Vijaya Keerthana**

M.Phil. Research Scholar, Dept of CS., Vivekanandha College for Women, Tiruchengode, TamilNadu, India

Assistant Professor, Dept of CS., Vivekanandha College for women, Tiruchengode, TamilNadu, India

**ABSTRACT:** To protect outsourced data in cloud storage against corruptions, adding fault tolerance to cloud storage, along with efficient data integrity checking and recovery procedures, becomes critical. The design and implement a practical data integrity protection (DIP) scheme for a specific regenerating code, while preserving its intrinsic properties of fault tolerance and repair-traffic saves. Cloud Data authentication: Data authentication ensures the group member that the data was accessed by a specified owner and the data was not altered en route. To provide these two functions, Dynamic Group key protocol relies on one trusted entity, KGC (Key Generation Center), to choose the key, which is then transported to each member involved. Each user is required to register at KGC for subscribing the key distribution service. The KGC keeps tracking all registered users and removing any unsubscribed users through revocation.

**KEYWORDS:** Data Integrity Protection, Key Generation Center, Cloud Storage.

## I.INTRODUCTION

One major use of cloud storage is long-term archival, which represents a workload that is written once and rarely read. While the stored data are rarely read, it remains necessary to ensure its integrity for disaster recovery or compliance with legal requirements. Since it is typical to have a huge amount of archived data, whole-file checking becomes prohibitive. Proof of retrievability (POR) and proof of data possession (PDP) have thus been proposed to verify the integrity of a large file by spot checking only a fraction of the file via various cryptographic primitives.
The outsource storage to a server, which could be a storage site or a cloud-storage provider. If we detect corruptions in our outsourced data (e.g., when a server crashes or is compromised), then we should repair the corrupted data and restore the original data. However, putting all data in a single server is susceptible to the single point of-failure problem and vendor lock-ins. As suggested a plausible solution is to stripe data across multiple servers. Thus, to repair a failed server, we can 1) read data from the other surviving servers, 2) reconstruct the corrupted data of the failed server, and 3) write the reconstructed data to a new server. POR and PDP are originally proposed for the single-server case. MR-PDP and HAIL extend integrity checks to a multiserver setting using replication and erasure coding, respectively. In particular, erasure coding (e.g., Reed-Solomon codes) has a lower storage overhead than replication under the same fault tolerance level.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

*(An ISO 3297: 2007 Certified Organization)*

**Vol. 3, Issue 12, December 2015**


**Figure 1.Cloud Storage Process**

## II. RELATED WORK

**NCCLOUD**: Applying Network Coding for the Storage Repair in a Cloud-Of-Clouds propose an implementable design for the functional minimum storage regenerating code (F-MSR), which maintains the same data redundancy level and same storage requirement as in traditional erasure codes (e.g., RAID-6), but uses less repair traffic.To implement a proof-of-concept prototype of NC Cloud and deploy it atop local and commercial clouds. We validate the cost effectiveness of FMSR in storage repair over RAID-6, and show that both schemes have comparable response time performance in normal cloud storage operations.

**F-MSR Implementation**

This system presents a systematic approach for implementing F-MSR. It specifies three operations for FMSR on a particular file object: (1) file upload; (2) file download; (3) repair. A key difference of our implementation from prior theoretical studies is that do not require storage nodes to have encoding capabilities, so our implementation can be compatible with today's cloud storage. Another key design issue is that instead of simply generating random linear combinations for code chunks, it also guarantees that the generated linear combinations always satisfy the MDS property to ensure data availability, even after iterative repairs. Here, to implement F-MSR as an MDS code for general (n,k).To assume that each cloud repository corresponds to a logical storage node.

## III. PROPOSED METHODOLOGY

### ALGORITHM IMPLEMENTATION

### AES ALGORITHM

High-level description of the algorithm

**Key Expansions**: Round keys are derived from the cipher key using Rijndael's key schedule. AES requires a separate 128-bit round key block for each round plus one more.

**Initial Round**: Add Round Key: Each byte of the state is combined with a block of the round key using bitwise XOR.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

*(An ISO 3297: 2007 Certified Organization)*

**Vol. 3, Issue 12, December 2015**

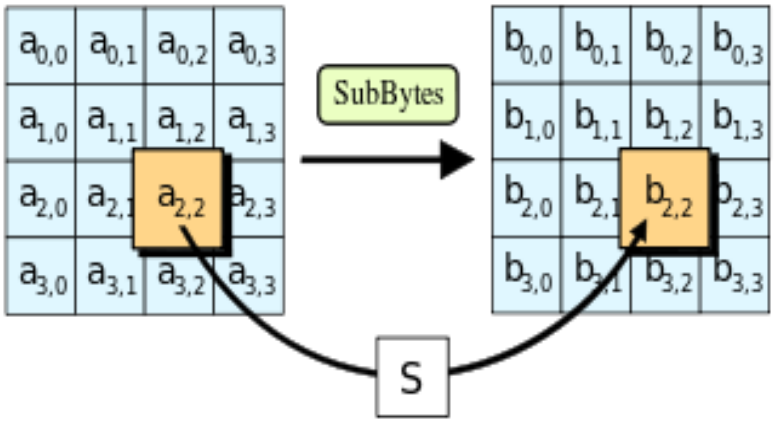
**Rounds**:

i) Sub Bytes: a non-linear substitution step where each byte is replaced with another according to a lookup table.

ii) Shift Rows: a transposition step where the last three rows of the state are shifted cyclically a certain number of steps.

iii) Mix Columns: a mixing operation which operates on the columns of the state, combining the four bytes in each column. Final Round (no Mix Columns).

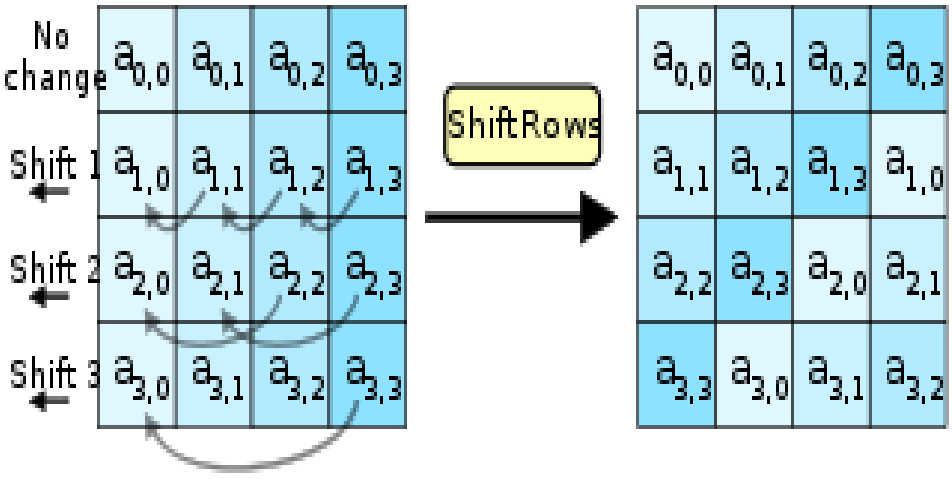
**THE SUB BYTES STEP**




In the Sub Bytes step, each byte in the state is replaced with its entry in a fixed 8-bit lookup table, S; $b_{ij} = S(a_{ij})$.

In the Sub Bytes step, each byte $a_{i,j}$ in the state matrix is replaced with a SubByte $S(a_{i,j})$ using an 8-bit substitution box, the Rijndael S-box. This operation provides the non-linearity in the cipher. The S-box used is derived from the multiplicative inverse over GF($2^8$), known to have good non-linearity properties. To avoid attacks based on simple algebraic properties, the S-box is constructed by combining the inverse function with an invertible affine transformation. The S-box is also chosen to avoid any fixed points (and so is a derangement), i.e., $S(a_{i,j}) \neq a_{i,j}$ and also any opposite fixed points, i.e., $S(a_{i,j}) \oplus a_{i,j} \neq 0\text{xFF}$.

**THE SHIFT ROWS STEP**



ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

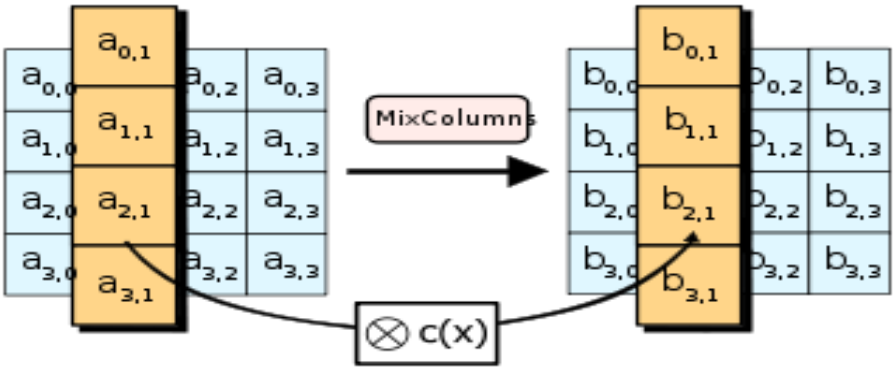
International Journal of Innovative Research in Computer and Communication Engineering

*(An ISO 3297: 2007 Certified Organization)*

**Vol. 3, Issue 12, December 2015**

In the Shift Rows step, bytes in each row of the state are shifted cyclically to the left. The number of places each byte is shifted differs for each row. The Shift Rows step operates on the rows of the state; it cyclically shifts the bytes in each row by a certain offset. For AES, the first row is left unchanged. Each byte of the second row is shifted one to the left. Similarly, the third and fourth rows are shifted by offsets of two and three respectively. For blocks of sizes 128 bits and 192 bits, the shifting pattern is the same.

**THE MIX COLUMNS STEP**




In the Mix Columns step, each column of the state is multiplied with a fixed polynomial *c(x)*.In the Mix Columns step, the four bytes of each column of the state are combined using an invertible linear transformation.

**FILE SHARING**

Any user in the group can store and share data files with others by the information sharing server. The encryption complexity and size of cipher texts are independent with the number of revoked users in the system. User revocation can be achieved without updating the private keys of the remaining users. A new user can directly decrypt the files stored in the information sharing server before his participation. A secure owner data sharing scheme. It implies that any user in the group can securely share data with others by the trusted information sharing server.

The proposed scheme is able to support dynamic groups efficiently. Specifically, new granted users can directly decrypt data files uploaded before their participation without contacting with data owners. User revocation can be easily achieved through a novel revocation list without updating the secret keys of the remaining users. It provides secure and privacy-preserving access control to users, which guarantees any member in a group to anonymously utilize the information sharing server resource. Moreover, the real identities of data owners can be revealed by the manager when disputes occur.

## IV. EXPERIMENTAL RESULT

The user can login through their username and password. After login the user can upload their data in cloud storage. If any modifications made by hackers in the uploaded data, the user can retrieve the original data from the server through this proposed system.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

*(An ISO 3297: 2007 Certified Organization)*

**Vol. 3, Issue 12, December 2015**

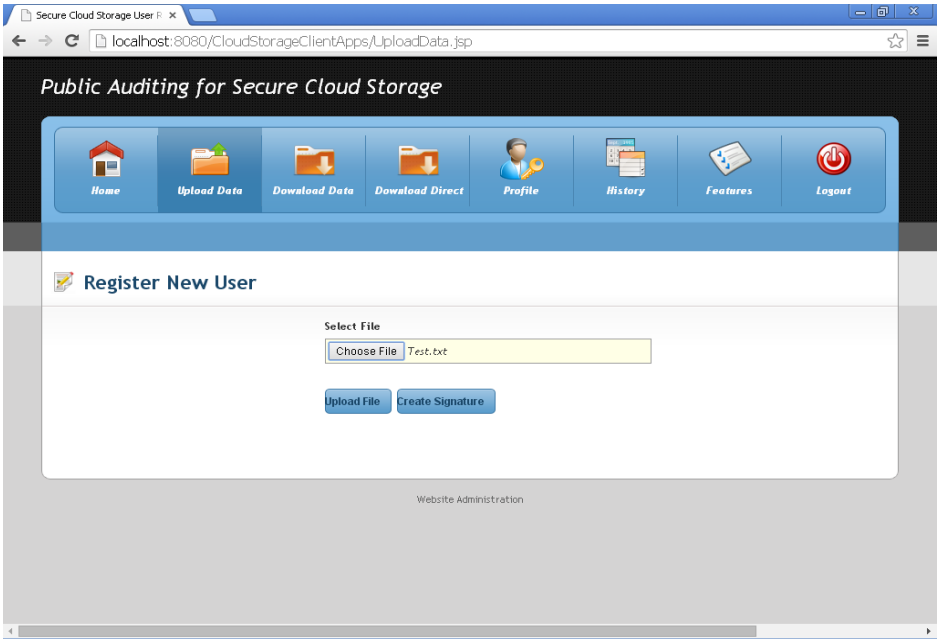



**Figure 2: Select the File to Upload and Create Signature Key Form**

The below form shows the original data that can be viewed by the user at the time he/she needs. It can be easily retrieved from the cloud storage by login through the user name and password as given to them earlier. The data's are in a safe manner and the user doesn't worry about their stored data in cloud server. The user not only store their data, they also share their data's to someone whom they wish.

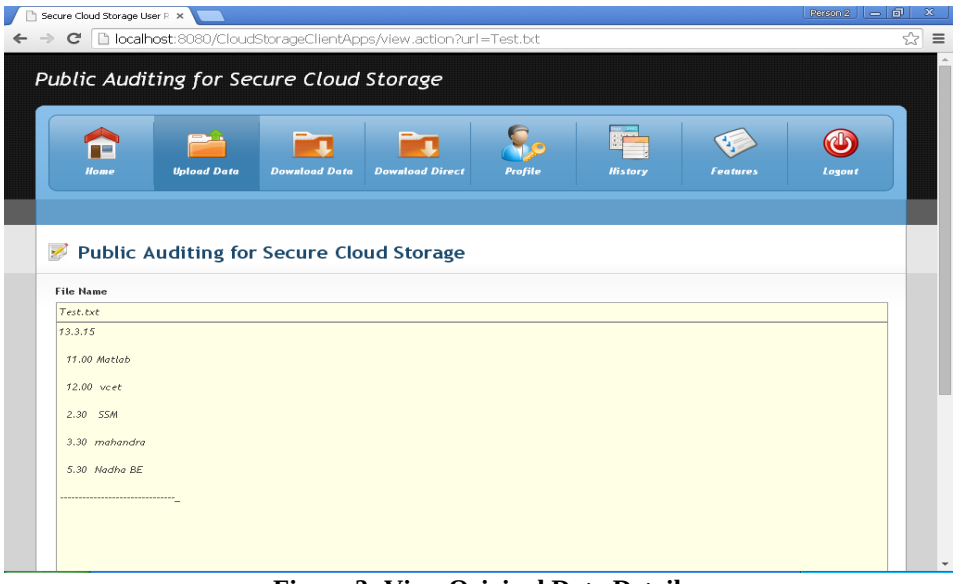



**Figure 3: View Original Data Detail**

## V. CONCLUSION AND FUTURE ENHANCEMENTS

### CONCLUSION

In this paper, to explore the problem of providing simultaneous public audit ability and data dynamics for remote data integrity check in Cloud Computing. Our construction is deliberately designed to meet these two important goals while

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

*(An ISO 3297: 2007 Certified Organization)*

**Vol. 3, Issue 12, December 2015**

efficiency being kept closely in mind. To achieve efficient data dynamics and improve the existing proof of storage models by manipulating the classic Merkle Hash Tree construction for block tag authentication. To support efficient handling of multiple auditing tasks, the further to explore a technique of bilinear aggregate signature to extend the main result into a multiuser setting, where TPA can perform multiple auditing tasks simultaneously. Extensive security and performance analysis show that the proposed scheme is highly efficient and provably secure.
Given the popularity of outsourcing archival storage to the cloud, it is desirable to enable clients to verify the integrity of their data in the cloud. The design and implementation of DIP scheme for the FMSR codes under a multi server setting. To construct FMSR-DIP codes, which preserve the fault tolerance and repair traffic saving properties of FMSR codes.To understand the practicality of FMSRDIP codes and analyze the security strength via mathematical modeling and evaluate the running time overhead via tested experiments.

## FUTURE WORK

In this project TPA (third part auditor) is used to access the auditing process of retrieve and edit the stored data in the cloud system. Many clients at a time will access the cloud server to store, retrieve and edit the data parallel. In that time single TPA may get overload to overcome this problem in future we will implement the multiple TPA processor when one TPA become overload other TPA will take process to auditing process.

## 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] H.C.H. Chen and P.P.C. Lee, "Enabling Data Integrity Protection in Regenerating-Coding-Based Cloud Storage," Proc. IEEE 31stSymp. Reliable Distributed Systems (SRDS '12), 2012.
[8] L. Chen, "NIST Special Publication 800-108," Recommendation for Key Derivation Using Pseudorandom Functions (Revised), http:// csrc.nist.gov/publications/nistpubs/800-108/sp800-108.pdf, Oct. 2009.
[9] R. Curtmola, O. Khan, and R. Burns, "Robust Remote Data Checking," Proc. ACM Fourth Int'l Workshop Storage Security and Survivability (StorageSS '08), 2008.
[10] 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.
[11] 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.