Data Integrity Protection for Cloud Storage through FMSR

Sonali V.Ghule
Department of Computer Science and Engineering
Walchand Institute of Technology, Solapur, Maharashtra, India

Pratibha S.Yalagi
Walchand Institute of Technology, Solapur, Maharashtra, India

Abstract- The use of cloud-based storage services are increasing rapidly and becoming an emerging trend in the data storage field. Cloud storage allows cloud clients to store their data. It reduces the burden of local data storage. Although cloud offers many facilities for its clients it cannot assure security of data stored there. When user store their data on the cloud, there may be a risk of losing the data, or sometimes data may be modified or updated. The modifications of data may occur by inside or outside attacker and due to this client may unable get back their data. In these cases often it is required to check for the integrity of data often. This paper proposes a new algorithm to provide data integrity assurance by striping the data across multi-servers and recovery procedures efficiently. During modifications our algorithm allows to retrieve a corrupted data without downloading whole file, thus minimizes the network traffic and time consumption. To protect integrity of the data, Data Integrity Protection (DIP) technique is designed and implemented for specific regenerating codes which are tested under different parameters and for efficiency remote integrity checking is integrated into regenerating codes and is deployed practically. Functional minimum-storage regenerating Data Integrity Protection (FMSR-DIP) scheme that selects the random subsets of the data from the server to check data integrity. The FMSR-DIP codes provide the facility of maintaining availability of the data.

Keywords – Cloud storage, Remote data integrity checking, regenerating codes

I. INTRODUCTION

Cloud storage offers by the cloud computing technology, which is a rapidly growing internet-based service with different benefits such as elasticity and low maintenance cost. It has many uses like data outsourcing service model, long term archival. The user focuses on the security of the data stored when it is outsourced to a third party. So it is desirable to allow the user to check the integrity when their data is corrupted accidentally or may be corrupted by third party. Hence it is necessary to ensure the data integrity through its entirety.

There are many solutions proposed for achieving data integrity in a cloud like POR (Proof of retrievability) [1] [2] and PDP (Proof of data possession) [3]. These two schemes were designed under a single server setting. The data was stored under a single server hence it leads to a single point of failure; it causes complete loss of our data. The single-server setting is extended to multi-server setting, in order to retain the data even after server failure. In multiple servers setting data is striped across multiple servers. If one of the servers gets failed then to repair a failed server, we can first read the data from surviving servers reconstruct the corrupted data of the failed server and finally write the reconstructed data to the new server. For this purpose, we use MR-PDP (Multiple Replica- Provable Data Possession) [4] and HAIL (High Availability and Integrity Layer) [5] integrity checking scheme. The limitation of these schemes is that the data is replicated across cloud which consumes more space and to retrieve the corrupted data whole file has to be accessed which violates the regenerating code design. Thus to overcome to these problems a new design data integrity protection scheme for regenerating codes have been proposed and implemented.

II. LITERATURE REVIEW

Cloud storage is used by the leading organizations to move data out of their own data centers and into the cloud. But cloud storage providers can present a risk to customers; namely, it becomes very expensive to switch storage providers. So they make a case for applying RAID-like techniques used by disks and file systems, but at the cloud storage level. They introduce RACS, a proxy that transparently distribute the storage load over many providers [6].

The Remote Data Checking Network Coding (RDC – NC),a novel secure and efficient RDC scheme for network coding based distributed storage systems [7]. Clients can check that data outsourced at untrusted remote servers remain intact over time by using Remote Data Checking (RDC) technique. RDC is acting as prevention tool that allow clients to periodically check if data has been corrupted or damaged and as a repair tool if damage has been detected. RDC developed initially in the context of a single server setting; but was later extended to store data redundantly at multiserver settings to verify or check the data integrity in distributed storage systems that rely on replication based system as well as on an erasure coding. Recently, a new technique was proposed to add the redundancy based on network coding, which offers tradeoffs because of its low communication overhead to repair corrupt or damaged servers. The Latent sector errors (LSE) refer to implement and experimentally show that it is computationally not expensive for both clients and servers [8]. LSEs are an important factor in data reliability, because a single LSE can lead to data loss when encountered during RAID reconstruction after a disk failure occurs. The LSEs happen at a significant rate in the welds, and are supposed to grow frequent with the new drive technologies and increasing drive capacities.

To ensure the integrity of data in cloud storage many efforts had been conducted. There are two fundamental approaches to client verification of data availability and data integrity [12]. These approaches are Provable Data Possession (PDP) and proof of retrievability (POR). Remote Data checking using PDP is one of the data integrity techniques. To allow the client that has stored data at an untrusted server to check that the server possesses the original data without retrieving it, they introduce a model for Provable Data Possession (PDP). Juels and Kaliski [1] proposed Proof of Retrievability scheme. To verify the data stored by user on remote storage in the cloud is not modified by the cloud Proof of retrievability scheme is used.

Proof of retrievability (POR) is a data integrity technique use for large files also. A POR technique allows a server to produce a proof that a client (verifier) can retrieve an intended file F, that is, the server retains and reliably transmits the file data required for the user to recover files F in its entirety. One of the techniques of cryptographic Proof of Knowledge (POK) is POR, but that is specially designed to handle a large file F. POR is the protocols in which the communication costs and the number of memory accesses is independent of the length of the F. In PDP based on a key generation algorithm, a client that has stored data on an untrusted server can verify that the server store the original data without retrieving it. While PDP based on the MAC is simpler than PDP based on a key generation algorithm. In PDP based on a key generation algorithm, the client generates pairs of matching keys public & secret key. While in PDP based on MAC, user generates message authentication code along with a set of secret keys. In POR based on encryption, few data bits per data block are encrypted instead of a whole file. This scheme is very useful for thin clients, because the data is not stored at client side. While in POR based on large file error correcting codes are employed to protect against corruption The PDP (Provable Data Possession) and POR (Proof of Retrievability) schemes useful for single server settings.

For multiserver settings MR-PDP and HAIL schemes are used. HAIL is the extension of the basic single server design of PORs. It does not suitable for the thin client [12]. HAIL allows storing the data on many servers; hence there is redundancy of the data. In HAIL scheme on the client side only small amount of data is stored.MR-PDP is the extension of the simple PDP scheme based on replication. In MR-PDP, there is a computation overhead because of replicas, but this is not the case in PDP. Erasure coding [10] is the technique used to store the data. When one of the servers get failing, to repair the failed server the whole file has to be accessed, which increase traffic and thus consumes more time. As a solution, our paper proposes a new and efficient scheme FMSR-DIP codes to provide integrity checking, fault and efficient recovery.

III. SYSTEM ARCHITECTURE

The following figure shows the flow of system architecture. The system architecture has three components. These components are a User, Web application and cloud storage providers.

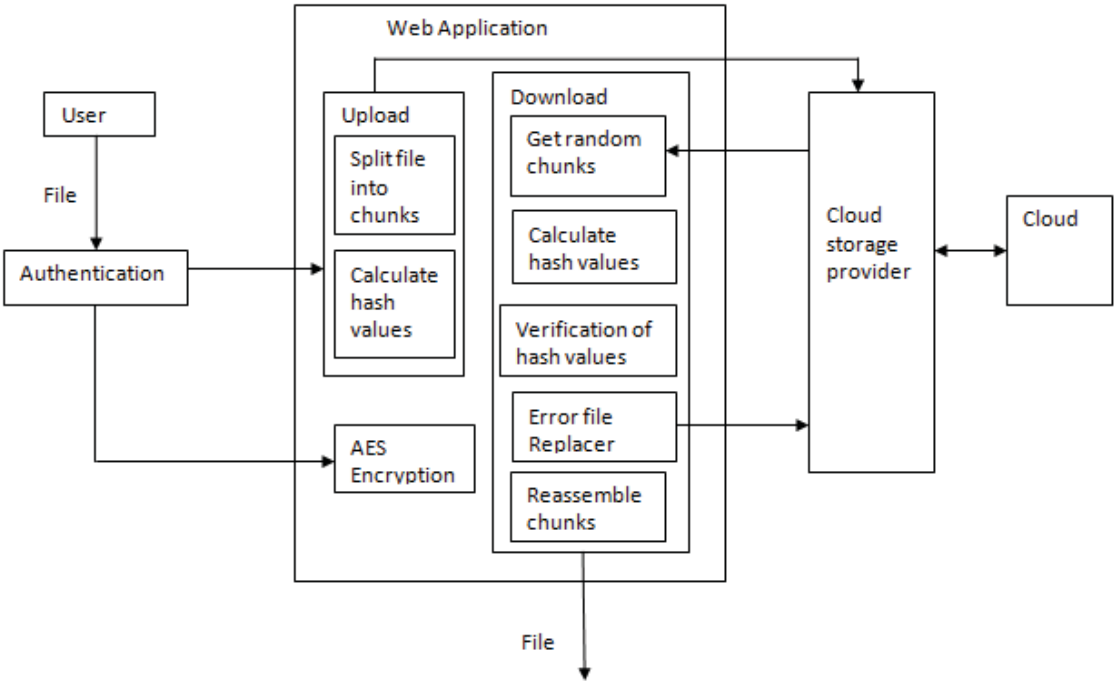

Figure 1. System Architecture

Web application acts as an interface between user and cloud. Here web application is NCCloud. NCCloud acts like a proxy server. NCCloud can support a variety of coding techniques, like the Functional Minimum Storage Regenerating (FMSR) codes [11]. To upload or download the files it will check for the authentication of users and after confirmation it will allow to proceed with the operations.

As shown in the diagram first user uploads the file to the cloud. Before uploading the file to the cloud it first divided into the chunks according to the server size. Metadata is created for each uploaded file. Metadata is a text file that contains file name, type of file, chunk numbers and one unique code for each chunk of the file. This unique code is Message Authentication Code (MAC). For each chunk one unique hash value will be generated. The file is stored in the cloud in encrypted format. User is able to download the file stored in cloud. Download operation checks the metadata file first and download the all chunks from the server. Download operation takes all the chunks randomly form the server. After that it will calculate hash value for each chunk. It checks the new calculated hash values with previous generated hash values that are stored in the metadata. If all values are same then it reassembles the all chunks and downloads the file. If one of the values will not get match then it will repair the file as per stored metadata value.

IV. PROPOSED WORK

There are problems in the previous data integrity checking schemes such as these are suitable for checking the data integrity of the data that are stored in single server settings. The new data integrity protection (DIP) scheme is used to overcome the problems that are present in the previous schemes. This new scheme of checking the data integrity for regenerating-coding-based cloud storage is integrated with FMSR (Functional minimum-storage regenerating) codes. This scheme provides the data integrity assurance. This scheme verifies the integrity of the small chunks by downloading the random rows from servers. The overview of this FMSR-DIP scheme is shown in the Figure 2.

A. Upload –

The first is the upload operation uploads the file to the cloud. Before uploading a file many processes like generation of secret keys, encoding are carried out for security purposes. When user select the file to upload it will be divided into multiple chunks. These chunks are encrypted before storing in the cloud. For each chunk hash value will be calculated. Message Authentication Code (MAC) are applied to recover the corrupted row and to protect the integrity of each row. Finally this information is appended to the metadata and metadata is used during repair operations.

B. Check –

Second is the check Operation to check the integrity of data which is stored. Here we are going to check whether the stored is corrupted or modified by inside or outside attacker. During upload operation metadata is created that contains the hash values. This hash values are used in this check operation.

C. Download –

Third is the download operation to download the file from servers. The download operation first checks the metadata files and downloads the chunks from servers and at the same time corresponding MACs are used to verify its integrity. The file will get download if the code chunks are not corrupted. If chunks are get modified by any unauthenticated person then before downloading the file it will get repair and then it will get download.

D. Repair –

Fourth is the repair operation which is similar to download operation and instead of downloading the k (n-k) chunk from any k servers during a server failure, one chunk is downloaded from each surviving server. Here n is the total number of servers and k is the any surviving servers from n servers. Finally the generated new chunks are encoded with the FMSR-DIP codes and are uploaded to a server and again metadata is updated.

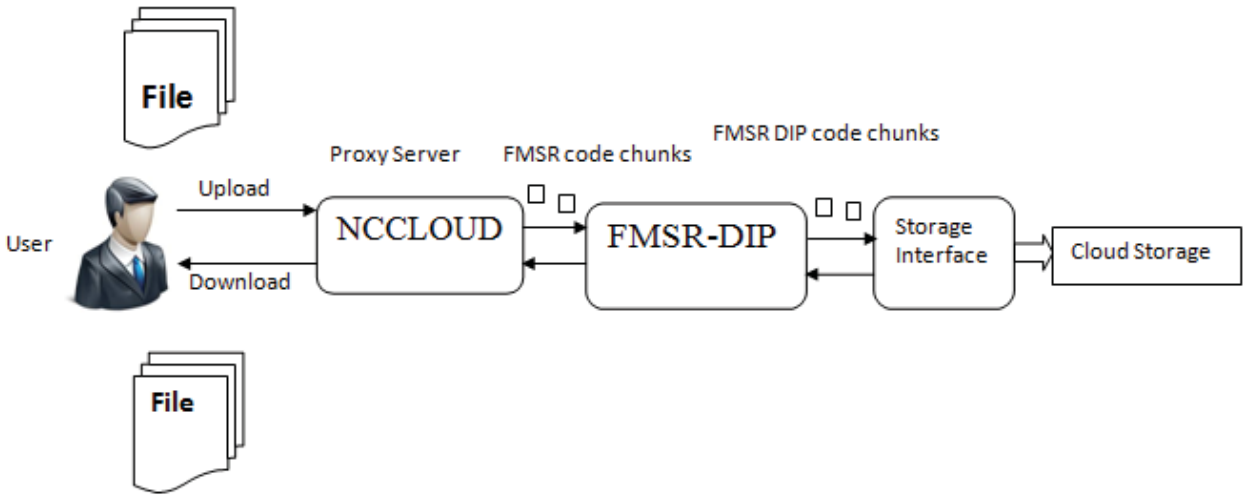

Figure 2. FMSR-DIP overview

V.CONCLUSION

A method for checking the integrity of stored data and the correctness of computations done by the cloud server is proposed. This scheme is introduced to reduce the computational overhead and storage overhead of the client. In this paper, we addressed about the FMSR-DIP codes to maintain the integrity of the data throughout its entirety. The cloud storage is popular due to its outsourcing archival storage service. Outsourcing the data in the cloud is very useful for clients and the clients need to verify the integrity of their data in the cloud. Though cloud computing offers many advantages, it also imposes security challenges which relate to cloud storage. The work preserves the properties of FMSR, the fault tolerance and repair traffic along with efficient integrity protection under multi-server setting. The data integrity technique described in this paper is more effective than the previous data integrity techniques. In previous data integrity techniques whole file is needed to be downloaded to repair the corrupted or modified file by unauthenticated person. But the technique that is proposed in this paper does not require downloading the whole file to repair the modified file. This method works only for static storage of data. The attacker or intruder can corrupt the images and videos. So as a future work we can focus on providing integrity protection to images and videos.

REFERENCES

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

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

[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] 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.

[5] 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.

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

[7] 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.

[8] B. Schroeder, S. Damouras, and P. Gill, "Understanding Latent Sector Errors and How to Protect against Them," Proc. USENIX Conf. File and Storage Technologies (FAST '10), Feb. 2010.

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

[10] I. Reed and G. Solomon, "Polynomial Codes over Certain Finite Fields," J. Soc. Industrial and Applied Math., vol. 8, no. 2, pp. 300-304, 1960.

[11] Y. Hu, 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.

[12] Sonali Ghule and Pratibha Yalagi, "A Comparative Study on Cloud Based Data Integrity Verification Schemes", International Journal of Computer Applications (0975 – 8887) National Seminar on Recent Trends in Data Mining (RTDM 2016)