Next-Gen Cloud Data Recovery: Harnessing Parity in Partially Distributed File Systems for Seamless Data Restoration

Anthani Kamala Priya[1], Shaik Jani[2*], Polamuri Sahithi[3], Anusha Darapureddy[4], Ravallakollu Madhuri[5]

[1,3] Assistant Professor, CSE, NS Raju Institute of Technology, Visakhapatnam, A.P, India.
[2]Assistant Professor, CSE, Vignan's Foundation of Science, Technology & Research, Vadlamudi, Guntur, A.P, India.
[4]Assistant Professor, CSE, Anil Neerukonda Institute of Technology & Sciences(A), Visakhapatnam, A.P, India.
[5]Assistant Professor, IT, Gokaraju Rangaraju Institute of Engineering & Technology, Hyderabad, A.P, India.

[1]priyaanits.it2007@gmail.com, [2*]skj.shaikjani@gmail.com, [3]sahithipolamuri1509@gmail.com, [4]anu.darapureddy@gmail.com, [5]rmadhuri580@gmail.com

Abstract. Cloud Computing provides towering benefits like huge scalability, low cost, accessible promptly still simultaneously it recommends distinct risks, burdens and vulnerabilities also [1]. Even though different cloud structure and services are emerging with vast expansion, some specific concerns stopped the organizations from completely joining the cloud due to various problem like security attacks, unavailability of data when he severs is attacked, more responding time, etc. In this, the work relates to a method or system that provides a parity distribution in a cloud storage system. This allows the identification of storage locations and retrieval of related chunks in distributed cloud storage by using variable chunk size, address of distributed chunk's locations and deciphering local & remote keys for the reconstruction of the required file separately which is under the attacked server.

Keywords: Cloud Storage System; Chunks; Address of Chunk; deciphering keys.

1 Introduction

It is a protocol developed as a modification on GFS/HDFS. It addresses there aspects of security [5] (confidentiality, Integrity, and Authentication) in data storage file distribution. Generally, a file stored in the cloud will be fragmented into to n chunks

K. R. Madhavi et al. (eds.), *Proceedings of the International Conference on Computational Innovations and Emerging Trends (ICCIET 2024)*, Advances in Computer Science Research 112,
https://doi.org/10.2991/978-94-6463-471-6_51

(see Figure 1) with addition data added by PDFSP protocol [2] and header and footer to each chunk of the file.

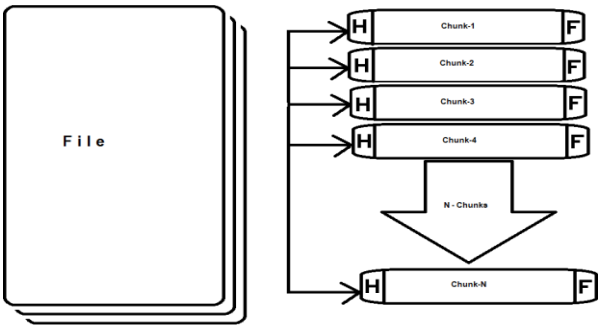


Figure-1: File Division into 'n' chunks

The Header of each fragment of the file (Chunk-N) accommodated with the following information as shown in Figure-2

- Local Key of 128 bit
- Remote Key of 128 bit
- Next Chunk Server Address
- Status Code of 128 bit.
- Audit Data of 1024 bits.

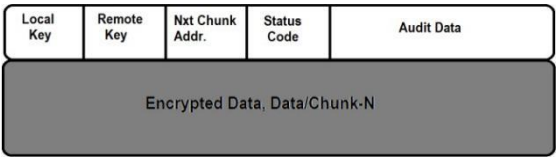


Figure-2: Fragment Header Structure

The Use Management Server (UMS) and Cloud Management Server (CMS) uses the header field of chunk to find its position and decrypt every file fragment (i.e. chunk) to integrate the chunks for the reconstruction to integrate the chunks for the reconstruction of original file by RFS [3]. Normally all the chunks are encrypted as well as decrypted with the private key. The private key is produced from a combination of three distinct keys retrieved from main server.

The first key is extracted with the help of previous chunk, the second key is the part of the remote key of the server (n) which holding (n-1) chunks, and the third key is obtained from Customer Client Machine (CCM), now all the three keys in addition to the local key of current server(n) are combined together to generate a deciphering key which is used for the decryption as displayed in Figure 3.

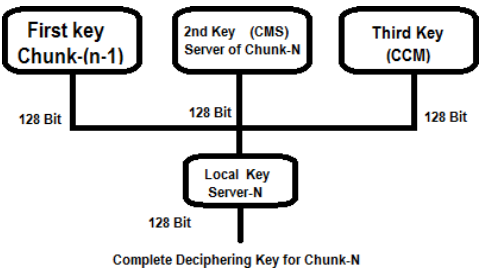


Figure-3: Deciphering Key Generation Process

In security the data integrity and availability will have utmost priority. The "Partially Distributed File System with Parity" implements the parity chunk in which the user can access the data under any condition of attack to the data in the cloud storage. Generally, in the cloud the data is fragmented into chunks, during the fragmentation the chunk may create a copy. in the original order of the normal chunks. So, if any action (power problems or security violation attack or denial of service) makes the server of the file unavailable. Now us system (parity) will come into the roll to recover the parity chunk which is equal to the chunk of the original file and rebuild the required data as original file [8]. So even if the file server is under attack the required data is made available to the request, which increases the characteristic of confidentiality and availability of data stored in the cloud.

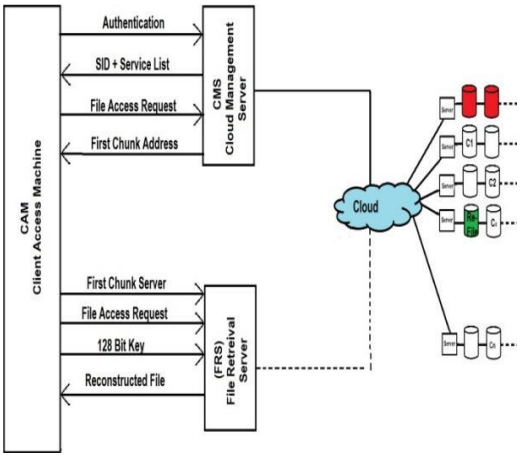
The Components of "Partially Distributed File System with Parity" are the following three: -

1. The Main Server i.e. Cloud Management Server (CMS)
2. The Computer for Customer i.e. Client Access Machine (CAM)
3. The Server for Keys, i.e. File Retrieval Server (FRS)

The Cloud Management Server is the main server retained and managed by the provider of the cloud storage; this regulates the operations. [4] The Client Access Machine is a personal computer which is enrolled by the provider of the cloud service as an administration machine for the use of user, this is done by hardware dangle supplied by the service provider to the user of the computer. It has complete control on the data where other machines and there will be limited access on the data by the public user, in some cases they no access right also. FRS is a server which uses the keys from local file server, CMS and CAM to reconstruct the original file from the chunks.

2 Methodology

The process of the "Partially Distributed File System with Parity" is show in the following Figure (4)



Fgure-4: Partial Distribution and Parity Process Flowchart

The customer utilizes the Client Access Machine to initiate an authentication and verification request/message, which is transmitted to the Cloud Management Server. Subsequently, the Cloud Management Server meticulously examines the request and grants approval. It then generates a session ID and furnishes a comprehensive list of available services associated with the server to the Client Access Machine [6]. Upon receiving this information, the user proceeds to forward a request for file access to the Cloud Management Server, triggering the commencement of the file access process. The primary objective at this stage is to retrieve all pertinent chunks of the requested file and meticulously reconstruct an exact replica of the intended file for the customer's use.

Throughout the process, the utmost care is taken to ensure the integrity and completeness of the file reconstruction. This involves meticulous retrieval and assembly of file components, ensuring that the resultant file is an accurate representation of the original. Once the customer has interacted with the requested file, performing actions such as reading, updating, or deleting, the individual file chunks are seamlessly reassembled into a new file within the cloud environment. This ensures that any modifications made by the customer are reflected in the stored version of the file, maintaining data consistency and integrity.

The complete File Access Process is as follows:
Step 1: The Client Access Machine (CAM) initiates an authentication request to identify the customer's computer, which is then forwarded to the primary server, the Cloud Management Server (CMS).

Step2: The Cloud Management Server (CMS) verifies the hardware details provided by the cloud service provider and authorizes access for the Client Access Machine (CAM). Additionally, the CMS sends the Session ID (SID) along with a comprehensive list of available services to the CAM for utilization.

Step3: The Client Access Machine (CAM) initiates a request to access specific data stored in cloud storage, communicating through the Cloud Management Server (CMS), even amidst an ongoing attack on the CMS.

Step4: The Cloud Management Server (CMS) provides the Client Access Machine (CAM) with the server address containing the initial chunk of the requested file.

Step5: The Client Access Machine (CAM) forwards a file retrieval request to the File Retrieval Server, including the 128-bit deciphering key, the (n-1) code part, and the address of the first data chunk.

Step6: The File Retrieval Server retrieves the first chunk of the specified file from both the Client Access Machine (CAM) and the Cloud Management Server (CMS), along with the necessary decryption keys.

Step7: The File Retrieval Server proceeds to access the next server in the sequence by utilizing the IP address obtained from the current processing server. This action enables the retrieval of the subsequent fragment or chunk of the required file.

Step8: The File Retrieval Server (FRS) continues its process by repeatedly retrieving the next file fragment, utilizing the decryption keys from the previous fragment, as well as support from the Client Access Machine (CAM) and Cloud Management Server (CMS), until the entirety of the file has been obtained.

Step9: FRS provides access to the rebuild file like the original file for CAM.

Step10: Client Access Machine now does all the required operations on the rebuild file and updates the file and submit the updated file to the FRS.

Step11: Now at FRS, the new version of updated file data is divided into modern-original chunks and stores the fragmented chunks with header and footer in various servers' randomly and finally it modifies the list available in the main server of cloud called Cloud Management Server [7] for the access by the customers latter.

3 Experimental Results

Experimental results demonstrate the effectiveness of the Partially Distributed File System with Parity Chunks (PDFSPC) in ensuring the availability, security, confidentiality, and integrity of data stored in cloud computing environments. By distributing file fragments across multiple servers and incorporating parity chunks, PDFSPC mitigates the risk of data unavailability caused by attacks or failures on individual servers. Through simulated attacks and failure scenarios, PDFSPC showcases its resilience in recovering

and reconstructing the required data even when a server or node is compromised. The experimental findings highlight PDFSPC's ability to dynamically rebuild primary files by utilizing parity chunks and distributing modified chunks across the cloud storage infrastructure. This ensures that authorized customers can access their data seamlessly, irrespective of potential disruptions or security breaches.

Certainly, presenting the experimental results in a tabular format can help in comparing them with related previous experimental results. Here's how we can structure the table:

Experimental Aspect	**Existed System Results**	**PDFSPC Results**
Data Availability	Moderate resilience against server failures	Enhanced availability through distributed parity chunks
Security	Vulnerable to attacks on Master Server	Robust against various attack vectors on Master Server
Confidentiality	Limited measures for safeguarding sensitive information	Ensures confidentiality through dynamic file rebuilding
Integrity	Basic data integrity mechanisms	Maintains integrity via distributed storage and parity
Scalability	Limited scalability in large-scale environments	Suitable for large-scale cloud storage systems
Real-world Deployment	Limited practical insights	Provides practical insights into real-world deployment

Table 1: Improvement of PDFSPC over Previous Results

This table provides a clear comparison between the experimental results of the Partially Distributed File System with Parity Chunks (PDFSPC) and related previous experimental results across various aspects such as data availability, security, confidentiality, integrity, scalability, and real-world deployment.

4 Conclusion

Furthermore, PDFSPC's robustness against various attack vectors on the Master Server, which traditionally represents a single point of failure, underscores its suitability for large-scale cloud storage systems. The experimental evaluation confirms PDFSPC's capability to maintain data availability while upholding stringent security measures, safeguarding the confidentiality and integrity of sensitive information stored in the cloud. Overall, the experimental results affirm PDFSPC as a viable solution for addressing the challenges associated with data availability, security, confidentiality, and integrity in cloud storage architectures, offering practical insights into its efficacy and resilience in real-world deployment scenarios.

References

1. Knorr E., Grumman G., “What Cloud Computing Really Means”, Info World
2. Security Analysis and Framework of Cloud Computing with Parity-Based Partially Distributed File System by Ali Asghary Karahroudy.
3. Evolution of Cloud Storage as Cloud Computing Infrastructure Service R. Arokia Paul Rajan1 , S. Shanmugapriyaa2, IOSR Journal of Computer Engineering 2012 www.iosrjournals.org
4. Madhavi, K. Reddy, A. Vinaya Babu, A. Anand Rao, and S. V. N. Raju. "Identification of optimal cluster centroid of multi-variable functions for clustering concept-drift categorical data." In Proceedings of the International Conference on Advances in Computing, Communications and Informatics, pp. 124-128. 2012.
5. K. Bicakci, D. D. Yavuz, S. Gurkan, "TwinCloud: Secure Cloud Sharing Without Explicit Key Management", 2016 IEEE Conference on Communications and Network Security (CNS), 2016.
6. Raju, S. Viswanadha, A. Vinaya Babu, G. V. S. Raju, and K. R. Madhavi. "W-Period Technique for Parallel String Matching." IJCSNS 7, no. 9 (2007): 162.
7. A Survey Paper on Data security in Cloud Computing, www.ijcseonline.org, 2016
8. Architecting the Cloud: Design Decisions for Cloud Computing Service Models (SaaS, PaaS, & IaaS), by Michael J. Kavis 2014.
9. Building theInfrastructure for Cloud Security by Raghuram Yeluri March 2014
10. Cloud Computing Protected: Security Assessment Handbookby John Rhoton Published 2013.