International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 08 Issue: 01 | January - 2024 SJIF Rating: 8.176 ISSN: 2582-3930

REMOVAL OF DUPLICATION IN CLOUD

Aditya Rajesh Narwade, Samadhan Shrawan Thube , Suraj Sudhakar Ingale, Saurav Pandit Mahajan, Prof. Kunal Ahire

[1] *adityanarawade003@gmail.com, Student of BE. Dept. of Information Technology MET-IOE, Nashik, India*
[2] *samadhanthube444@gmail.com, Student of BE. Dept. of Information Technology MET-IOE, Nashik, India*
[3] *surajingale554@gmail.com, Student of BE. Dept. of Information Technology MET-IOE, Nashik, India*
[4] *sauravmahajan2002@gmail.com, Student of BE. Dept. of Information Technology MET-IOE, Nashik, India*
[5]*Internal Guide Dept. of Information Technology MET's Institute of Engineering, Nashik, India*

Abstract - Deduplication involves eliminating duplicate or redundant data to reduce stored data volume, commonly used in data backup, network optimization, and storage management. However, traditional deduplication methods have limitations with encrypted data and security. The primary objective of this project is to develop new distributed deduplication systems that offer increased reliability. In these systems, data chunks are distributed across the Hadoop Distributed File System (HDFS), and a robust key management system is utilized to ensure secure deduplication with slave nodes. Instead of having multiple copies of the same content, deduplication removes redundant data by retaining only one physical copy and referring other instances to that copy. The granularity of deduplication can vary, ranging from an entire file to a data block. The MD5 and 3DES algorithms are used to enhance the deduplication process. The proposed approach in this project is the Proof of Ownership (POF) of the file. With this method, deduplication can effectively address the issues of reliability and label consistency in HDFS storage systems. The proposed system has successfully reduced the cost and time associated with uploading and downloading data, while also optimizing storage space.

Key Words: *Cloud computing, data storage, file checksum algorithms, computational infrastructure, duplication.*

1. INTRODUCTION

Cloud computing is an efficient technology for storing large amounts of data that can be easily accessed from anywhere at any time, eliminating the need for expensive hardware, dedicated space, and software maintenance. Meeting the growing demand for storage is a complex and time-consuming task that requires extensive computational infrastructure to ensure successful data processing and analysis. As the number of users and the size of their data continue to grow exponentially, data deduplication becomes increasingly essential for cloud storage providers. By storing a single copy of duplicate data, cloud providers can significantly reduce their storage and data transfer costs. This project provides an overview of cloud computing, cloud file services, accessibility, and storage, while also examining storage optimization through deduplication. It explores existing data deduplication strategies, processes, and implementations to benefit both cloud service providers and users. Additionally, the project proposes an efficient method for detecting and removing duplicates using file checksum algorithms, which requires less time than other pre-implemented methods.

A. Advantages Of Cloud Storage

Compatibility: All of the cloud storage providers we've looked at so far feature desktop folders on all of their platforms. Users may move files between their local storage and the cloud using this method. Bandwidth: Instead of emailing files to someone, you may send a web link to them via email.

Accessibility: Files stored on the server may be viewed from any computer with an Internet connection.

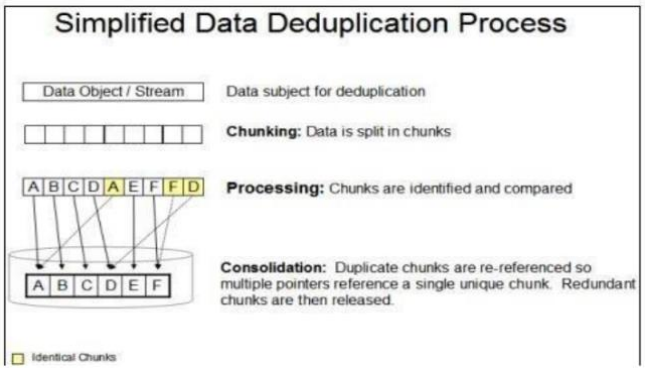
Recovering from disaster: It is strongly advised that firms have a backup strategy in place at all times. Businesses may utilize cloud storage as a backup strategy by storing a second copy of crucial files. These files are kept on a distant server and may be viewed using an internet connection.

B. Problem Statement

The system contains three entities: 1. Data owner (Ui) who uploads, downloads the data and is willing to delete the data that is stored into the cloud server 2. Cloud Server which provides the data storage services3. Key management server which stores and manages the keys associated with the storage.

C. Overview Of Data Deduplication

Any cloud-based data security solution, for instance, must be willing and able to provide services at a cheaper rate to intending customers compared to what they can sum up or create on their own. Data deduplication among numerous clients is one way being utilized to achieve this aim, with the number of paying clients. Because there are several ways for eliminating duplicate data, the variations in the solutions, as well as the influence they may have on security and the rate at which data is being transmitted must be understood by service providers and their consumers.



D. Chunking Of Data Stream

A technique for identifying redundancies by separating data into predetermined pieces can be referred to as chunking. It depends on the deduplication process's technology and location; these units might be files or more detailed elements like blocks. When compared to other ways, file-level deduplication is less adaptable. If the deduplication system is aware of the features, it may be able to distinguish specific components inside specific files. The many ways in which data can be chunked are listed in the table below. The data deduplication ratio is affected by each approach.

2. PROPOSED METHODOLOGY

The proposed system mainly includes four phases namely,

- Preparing Tag for upload
- Segmenting the file and storing into cloud
- Key sharing
- Decryption

Preparing Tag for upload

Clients register to the Cloud server with their data and login the page for transferring the document. User selects the file for upload to cloud server, the Cloud server stores the file and file level deduplication is checked. Tag is generated using message digest (MD5) algorithm for producing the required hash value.

Segmenting the file and storing into cloud

We produce united keys for each block separately for block level de-duplication. Here we give a filename and secret key for file approval in future. The individual blocks are encrypted by 3DES algorithm. The original text is encoded three times with convergent keys. So when decoding the original content it requires the same key.

Key Sharing

After encryption the convergent keys are securely presented to the key management systems. Key organization slave checks duplicate copies of centered entries in KMCSP. Key Management systems keeps up CSV archive to check confirmation of verified and stored keys securely. The customers share the basic keys are suggested by their own specific proprietorship (proof of ownership). In the event that User required to do erasure, absolutely he should demonstrate evidence of possession to delete his own contents.

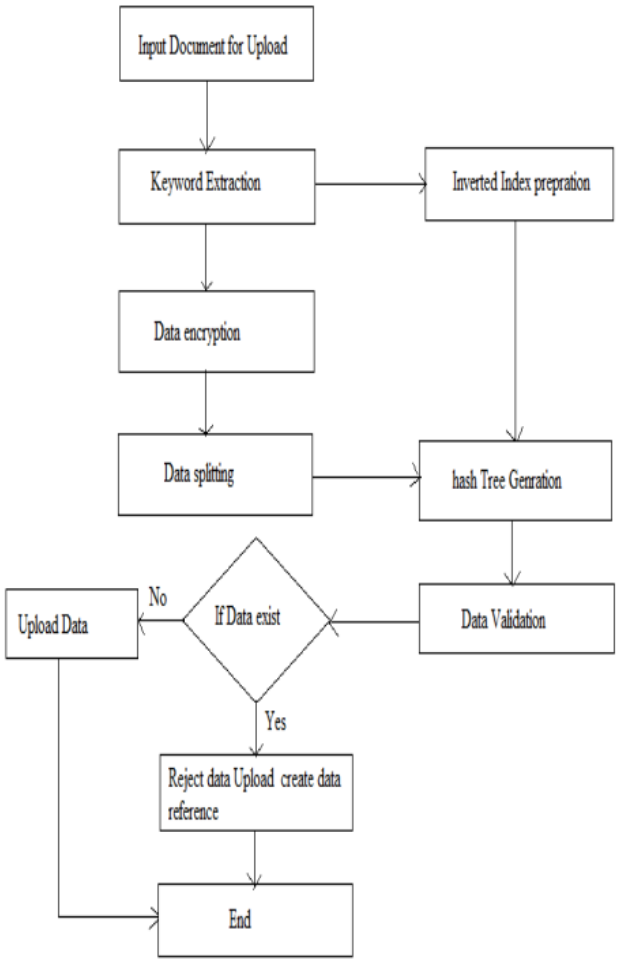


Fig. Flow of the proposed system

3. PROJECT METHODOLOGY

A. Algorithm For The System

The file checksum technique is used in this project to gauge the file's hash value as the primary prerequisite for removing duplicates. The method takes a random message as input and returns a 128-bit "message digest" of it. It is considered to be practically impossible to produce two messages with a matching message digest, or a message with a pre-specified goal message digest. The MD5 algorithm was created for digital signature systems where a large file must be safely "compressed" before being encrypted with a private (secret) key using a public-key cryptosystem.

B. File Checksum

Ways to assess hashes and checksum to eliminate duplicates from a selection were demonstrated in File Checksum. Data duplication removal is examining storage devices, such as a hard drive or a server, seeking duplicate data, and selectively removing them. While the following material discusses picture files, any file type on a computer may be opened using the same approach. This is because picture files take up a lot of space, and being able to remove all but one instance might drastically reduce the amount of storage required, especially in circumstances where there isn't a stringent file handling protocol in place when adding files. The process of using an

algorithm to confirm the validity of a computer file, which is included in the file checksum, is known as file verification.

C. File Verification

The process of using an algorithm to confirm the validity of a computer file, which is included in the file checksum, is known as file verification. This may be implored by intermittently comparing two files, but it requires two copies of the same file and may ignore systemic errors that impact both files (Morris 2015). A file's integrity can be endangered, causing the file to become corrupted. A file can be destroyed in a variety of ways, including bad storage media, transmission problems, copying or relocation errors, and software flaws. Comparing the hash value of a file to a previously determined value, verification that is hash-based gives the assurance that the file is free from errors. The file is assumed to be unaltered if there's a match in the value. ".sha1" extension denotes a checksum file in sha1sum format that contains 160-bit SHA-1 hashes. A checksum file with a ".md5" extension, contains 128-bit MD5 hashes in md5sum format. A checksum file with the suffix ".sfv" contains 32-bit CRC32 checksums in a basic file verification method.

D. Checksum Algorithm

Parity byte (also known as parity word):

The most basic checksum technique is the check for parity over time, it breaks data into n-bit "words"each, and The exclusive or (XOR) of all the other terms is then computed. As an additional word, the result is attached to the message. The recipient evaluates the exclusive of all the message's words, including the checksum, to assess the message's integrity; if the output is not a term consisting of n zeros, the recipient knows there was an error in transmission. Any transmission fault that flips a small piece of the message, or an odd number of bits, is identified as an invalid checksum by this checksum. The probability of a two-bit error being undetected is 1/n if the damaged bits are picked at random.

Modular Sum:

The checksum is computed by adding all the "words" as numbers that are unassigned, eliminating any extra bits, and multiplying the total by the complement of the two. The recipient repeats the process, including adding all of the words in the same sequence, to verify the message. An error has occurred if the checksum does not return a word full of zeros. This version also detects single-bit faults, but not multiple-bit faults.

Message Digest:

MD5 File Verification is a program that can generate a checksum file. Checksums will be utilized. The main purpose is to collect all checksums for the collection of images from where we would like to erase repetitions, sort them by checksum values, and then rapidly locate any duplicates using a spreadsheet calculation technique. We may use the spreadsheet as a checklist to delete duplicate files if there are any.

E. Md5 Checksum Algorithm

This algorithm also known as the algorithm of MD5 message-digest collects messages of any length as input and creates a 128-bit message digest. It is thought to be computationally impossible to produce two messages with a matching message digest, or any message that the message digest has known beforehand.

We begin by assuming that we have an n-bit message and that we wish to extract its message digest. n is a non- negative arbitrary integer that is capable of being "0", not a multiple of "8", and be any size. We picture the following parts of the message being written down: m_0 m_1 ... m_{n-1}

To calculate the message digest, complete the following five steps.

Step 1: Attach the Padding Bits

The message is "stretched" (padded) to a length of 448 modulo 512 bits. In other words, the message is just 64 bits long, falling short of being a multiple of 512 bits. Padding is often executed, even if the length of the message is already equal to 448, modulo 512. Padding is accomplished by attaching a single "1" bit to the message, followed by "0" bits until the length of the stretched message in bits equals 448 modulo 512bits. A total of 512 bits is added, with a minimum and maximum of 1 bit.

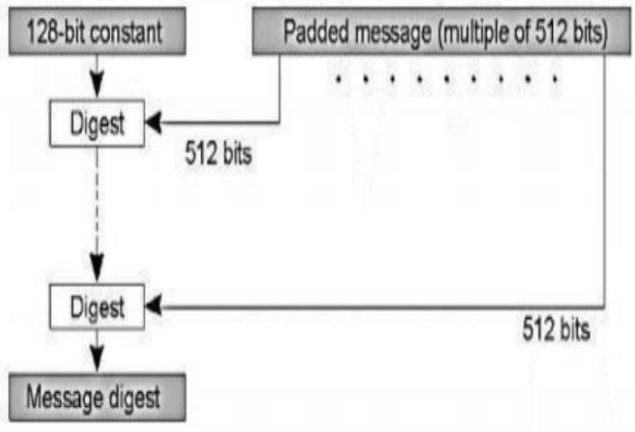
Step 2: Add the Length

A 64-bit representation of b is added to the result of the previous step. If b is larger than 264 in this case that it is, just the low-order 64 bits of b are used. in which these bits are joined as two 32-bit words, low order word first, as is customary. At this point, the final message's length (after padding with bits and b) is an exact multiple of 512 bits. To put it another way, the message's length is a multiple of 16 (32-bit) words. M[0... N-1], noting that N is a multiple of 16.

Step 3. Initialize MD Buffer

The message digest is computed using a four-word buffer (A,B,C,D). Each of the registers A, B, C, and D is a 32-bit register. In hexadecimal, these registers are assigned to the following values (low-order bytes first):

Word A: [01] [23] [45] [67];
Word B: [89] [ab] [cd] [ef] ;
Word C: [fe] [dc] [ba] [98] ;
Word D: [76] [54] [32] [10];



4. GUI/WORKING MODULES

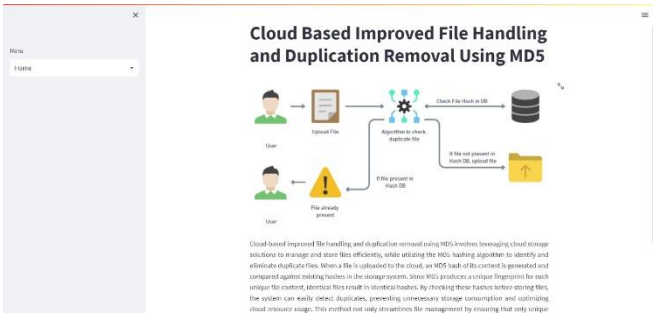


Fig4.1 : Home Page

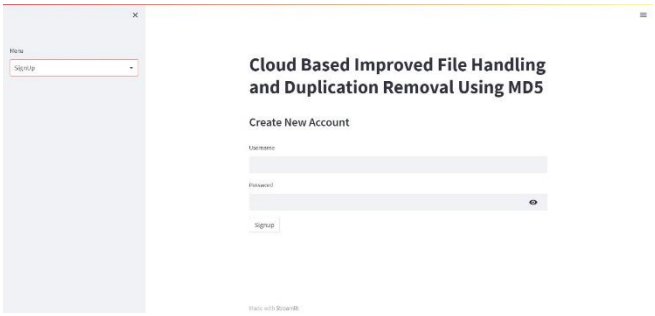


Fig4.2 : Login Page

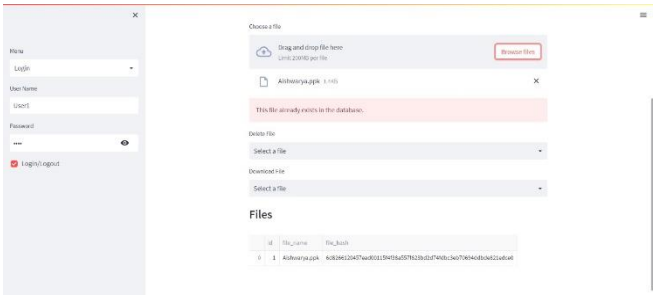


Fig4.3 : User Dashboard

Fig4.3.4: Pie Chart

5. CONCLUSIONS

The web application has been designed to efficiently identify, share, and eliminate duplicate data in the cloud using file checksum. It offers valuable support for organizations engaged in highly repetitive operations that involve frequent data copying and storage for future reference or recovery purposes. This approach forms a critical part of the backup and disaster recovery solution, enabling enterprises to store data repeatedly and facilitate swift, reliable, and cost-effective data recovery. For example, when a file is backed up on a weekly basis, it generates a significant amount of duplicate data, consuming a considerable amount of disk space.

Employing file checksum for data duplicate removal involves conducting an analysis to eliminate these sets of duplicate data, retaining only unique and essential information, thereby freeing up storage space. The primary challenge was to ensure that all files stored in the database did not contain duplicates of themselves, which was addressed by utilizing PHP (Hypertext Pre-processor) software. The educational impact of this system lies in the development of a new and efficient method for identifying, sharing, and removing data duplicates using file checksum in the cloud.

6. REFERENCES

[1] D. Meister, A. Brinkmann, ”Multi-level comparison of data deduplication in a backup scenario”, Proceedings of SYSTOR 2009: The Israeli Experimental Systems Conference, ACM, pp. 8:1-8:12, 2009.

[2] Daniel J. Abadi, Data Management in the Cloud: Limitations and Opportunities, IEEE Data Engineering Bulletin, Volume 32, March 2009, 3-12.

[3] Edgar J. Codd, 1990; The Relational Model for Database Management Version 2; Addison Wesley Longman Publishing Co., incBooston, MA, USA ISBN:0-201-141-14192-2.

[4] Edwin Schouten; 2012; Cloud Computing Business Benefits; http://wired.com/insights/ 2012/10/5-cloud-business-benefits/

[5] Ames B., Rajkman B., Zahir T., 2009 MetaCAN: HanessingStorgae Clouds for High Performance Content Delivery; Journal of Network and Computer Application, 1012-1022, 2009.