isrjournals

International Journal of Advanced Research in Computer Science Engineering and Information Technology

Volume: 6 Issue: 3 Apr,2024,ISSN_NO: 2321-3337

DYNAMIC DETECTION OF DUPLICATE CONTENT IN CLOUD USING MERKLE HASH TREE

Chakravarthi.A.S[1], Karthik.K[2], Dilli Prasad.K[3], Aarthi V[4*]

[1234]UG Scholar-Dept.CSE, GRT Institute Of Engineering & Technology, Tiruttani, India.

[4*] Asst. Professor-Dept.CSE, GRT Institute Of Engineering & Technology, Tiruttani, India.

chandhuas2085@gmail.com , karthikkatta4526@gmail.com ,dilliprasad601@gmail.com

***Corresponding Author: aarthi.v@grt.edu.in**

Abstract

We all know that Cloud computing is being used all the companies for effective data storage and retrieval system. Cloud has to be so scalable to handle the load from the users. But the storage of cloud would become a problem as number of users and data storage will be infinite. Mainly duplication of files would be the main issue. There is no process to filter and remove the same files from the cloud server in order to avoid the storage issues. In our project, we are removing the duplicate files from the server through Merkle hash Tree Algorithm and allows to store only the Non duplicate files in the cloud server. This will surely avoid Deduplication of files in Cloud server.

Keywords: *Data storage, Duplicate files, Storage issues, Merkle hash Tree Algorithm, Deduplication, Cloud server, Data retrieval, Effective system.*

1. Introduction

Cloud computing has emerged as a crucial technology in the IT industry, offering reliable software, hardware, and Infrastructure as a Service (IaaS) via the Internet and remote data centers. It facilitates complex computing tasks across various IT functions, from storage to application services, attracting organizations and individuals due to reduced capital costs and the increasing volume of data. Cloud service providers integrate frameworks for parallel data processing, enabling users to access resources and deploy programs effortlessly. With its model of ubiquitous, on-demand access to computing resources, cloud computing addresses economic and technological barriers, allowing organizations to focus on core business activities while enjoying flexibility and resource availability. Cloud service models encompass Platform as a Service (PaaS), Software as a Service (SaaS), and IaaS. The rise of wireless networks and mobile devices has fueled mobile cloud computing, allowing users to outsource tasks and store data externally, improving performance and user experience despite inherent limitations.

2. Related Work

In contemporary enterprise data protection, disk-based deduplication storage has superseded tape libraries, offering a compact and economical solution. Deduplication, the process of eliminating redundant data segments, facilitates efficient storage on disk. Crucially, high throughput, exceeding 100 MB/sec, is imperative for swift backup completion. However, achieving this on low-cost systems with limited RAM poses challenges, necessitating innovative solutions. This paper presents three techniques utilized in the Data Domain deduplication file system to alleviate disk bottlenecks: the Summary Vector for segment identification, Stream-Informed

Segment Layout for optimized data arrangement, and Locality Preserved Caching for enhanced cache hit ratios. These techniques significantly reduce disk accesses, enabling a modern system to operate at 90% CPU utilization with minimal hardware requirements. Ultimately, they empower efficient single-stream throughput of 100 MB/sec and multi-stream throughput of 210 MB/sec, ensuring robust data protection in enterprise environments. [1]

Imagine having lots of files on your computer that are pretty much the same. We looked at a bunch of these files from different places around the world where many people work. From what we learned, we made a new way for computers to store these files more efficiently. We figured out how to make the files take up less space while still keeping all the important stuff. We also made sure our method didn't use up too much computer memory, CPU power, or hard drive resources. Our new system helps computers handle big amounts of data better without slowing down. We explain how it works and show that it does a good job at saving space and organizing the files.[2]

We looked at the files saved on 857 computers at Microsoft over four weeks. We wanted to see how good a method called data deduplication is at saving space on these computers. We compared two ways of doing this: one where entire files are checked for duplicates, and another where only parts of files are checked. We found that checking entire files saves about three-quarters of the space compared to the more thorough method for regular files, and 87% for backup files. We also checked how messy the files were and found that most of them are not messy. Additionally, we looked at how big the files were and found that most of them are large and not organized.[3]
Many companies use deduplication to save money and make their data centers more efficient. But making this process faster for important tasks has been tricky because it usually slows things down. Our solution, called iDedup, tackles this problem. We noticed that when you organize data in a certain way, you can save space and make things faster. So, we only organize certain chunks of data to avoid making things messy. We also found a clever way to keep track of what's what without slowing things down too much. Our tests show that iDedup can save a lot of space without making computers much slower.[4]

Studies have found that Cloud storage systems often have a lot of duplicate data, especially when it comes to small tasks. However, just removing this duplicate data can make things messy and slow. We came up with a solution called POD that focuses on making things faster without sacrificing space savings. Unlike other methods that focus mainly on saving space, POD also considers making small tasks faster. Our tests showed that POD can make Cloud storage systems perform up to 87.9% better than other methods, with an average improvement of 58.8%. Plus, it still saves space just as well as other methods, making it a win-win solution.[5]

3. Objective

The project aims to enhance Dropbox cloud storage by implementing a scalable and efficient system for detecting and removing duplicate files during the upload process. This system will cater to multiple users and device, the project seeks to quantify the potential storage space savings that can be achieved through the proposed method in real-world scenarios, providing valuable insights into the practical benefits of duplicate file detection and removal in cloud storage environments.

4. Proposed System

Data owner will upload the file, it will encrypt and store in cloud. All deduplicate files are maintained in separate server with hashing index numbers, file which is frequency accessed (reusable) by user is maintained in a separate server and finally unused files are maintained in a separate server. We will be storing both Data & the videos in the server For Data Bucketization, Data is split into smaller Parts, Encrypted and stored along with the Index file. Requested Data is compared with the Index file and Data is retrieved. The Videos are Split into smaller Chunks based on the time Frames. User will request a Video along with the time Frame. Server will Stream the Video from the Requested Time Frame of the User.

5. Architecture Diagram

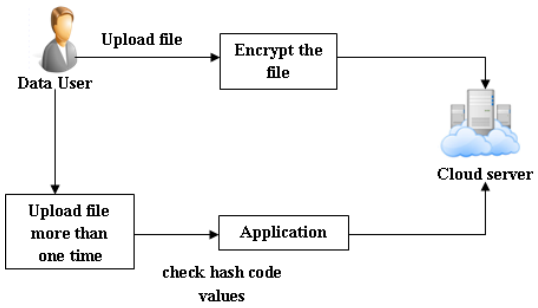

Fig: 5.1 Architecture Diagram

6. Algorithm

A Merkle hash tree, or simply Merkle tree, is a data structure used in computer science to efficiently summarize and verify the integrity of large sets of data. It plays a significant role in blockchain technology, peer-to-peer networks, and other systems that require secure and efficient data verification. Here's a breakdown of how the Merkle hash tree algorithm works.

7. Implementation

A modular design reduces complexity, facilities change (a critical aspect of software maintainability), and results in easier implementation by encouraging parallel development of different part of system. Software with effective modularity is easier to develop because function may be compartmentalized and interfaces are simplified. Software architecture embodies modularity that is software is divided into separately named and addressable components called modules that are integrated to satisfy problem requirements.

7.1 User Interface Design

In this Module User interface is created so that the data owner will upload the data to the server. The main objective of this module is to store and share the data by uploading the file to the remote machine. Data is Hashed and applied XOR functionality and then finally stored in the main server.

7.2 Cloud Server

Data owner will upload their data to the cloud server and request for a particular file is send to cloud server. Both the upload and the file request are handled by the main Cloud Server. During the file request is processed main server will communicate with the data owner and the files are retrieved only after the approval given the data owner.

7.3 MHT-Merkle Hash Tree

In this module, data is encrypted, Hashed and XOR is applied only then the data is uploaded to the server. Once the data is uploaded, the entire data is chunked into multiple parts using Merkle Hash Tree algorithm and stored in separate data servers. In cryptography and computer science, 2a hash tree or Merkle tree is

a tree in which every leaf node is labeled with the hash of a data block and every non-leaf node .

7.4 Deduplication

In this module user will upload more number of file on cloud on the same time many people will upload same file in different file name. So that more number of space will occupied in cloud. For that we implement duplicate detection by reading the content of the text file. And also we separate the file by frequency accessing and infrequent file. We will maintain the keyword in index in encrypted form.

8. Experimental Results

Fig 8.1 User Registration

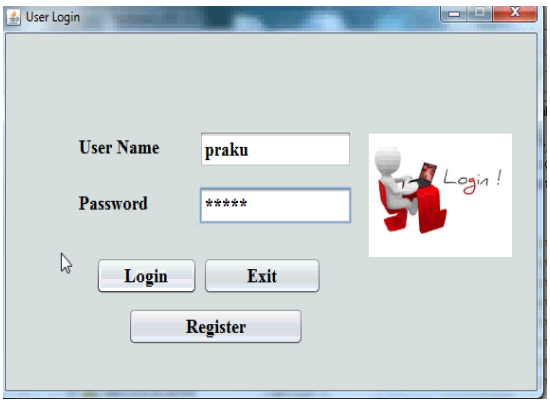

Fig 8.2 Login page

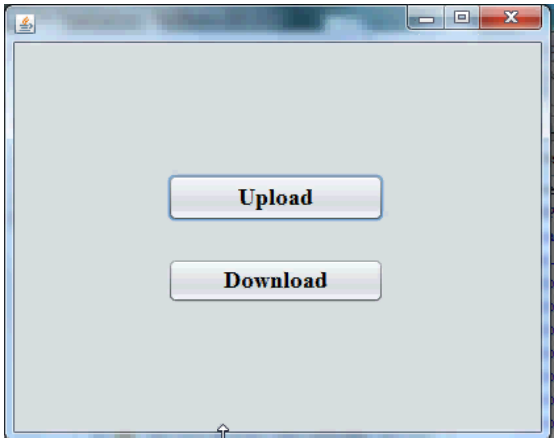

Fig 8.3 Upload & Download File

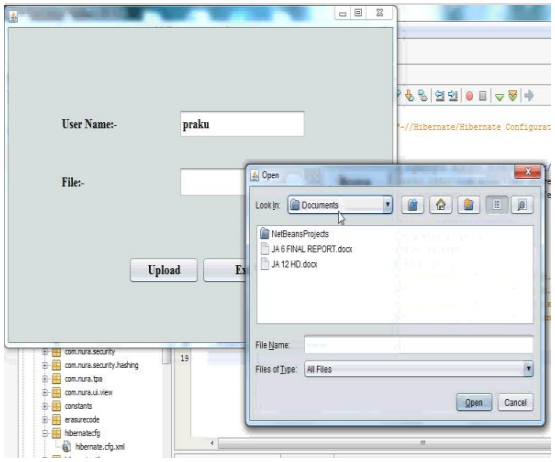

Fig 8.4 Upload File

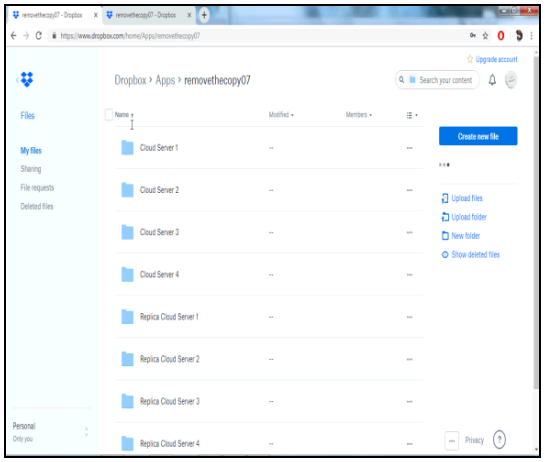

Fig 8.5 Cloud Server

9. Conclusion & Future Work

Thus the project infer that user data will secure on cloud by splitting the data on cloud with encrypt the data. Whenever user access the file from cloud , it will reconstruct the file and provide it to the user. Through this paper we found that maximum level of duplication of files will avoid using our techniques. Apart from Data / Text files PDF, Videos, Images are also to be verified and deduplications are identified. Multiple Clouds to be integrated if one cloud is full then automatically data are transferred to the another clouds and the same process.

10. References

[1] B. Zhu, K. Li, and R. H. Patterson, “Avoiding the disk bottleneck in the data domain deduplication file system,” in Proc. 6th USENIXConf. File Storage Technol., 2008, pp. 1–14.

[2] A. El-Shimi, R. Kalach, A. Kumar, A. Ottean, J. Li, and S. Sengupta, “Primary data deduplicationlarge scale study and system design,” in Proc. USENIX Annu. Tech. Conf., 2012, pp. 285– 296.

[3] D. T. Meyer and W. J. Bolosky, “A study of practical deduplication,” ACM Trans. Storage, vol. 7, no. 14, pp. 1–20, 2012.

[4] K. Srinivasan, T. Bisson, G. R. Goodson, and K. Voruganti, “iDedup: Latency-aware, inline data deduplication for primary storage,” in Proc. 11th USENIX Conf. File Storage Technol., 2012, pp. 1–14.

[5] B. Mao, H. Jiang, S. Wu, and L. Tian, “POD: Performance oriented I/O deduplication for primary storage systems in the cloud,” in Proc. IEEE 28th Int. Parallel Distrib. Process. Symp., 2014, pp. 767–776.

[6] A. Wildani, E. L. Miller, and O. Rodeh, “Hands: A heuristically arranged non-backup in-line deduplication system,” in Proc. IEEE 29th Int. Conf. Data Eng., 2013, pp. 446–457.

[7] J. An and D. Shin, “Offline deduplication-aware block separation for solid state disk,” in Proc. 11th USENIX Conf. File Storage Technol., 2013, pp. 1–2.

[8] C. Constantinescu, J. Glider, and D. Chambliss, “Mixing deduplication and compression on active data sets,” in Proc. Data Compression Conf., 2011, pp. 393–402.

[9] V. Tarasov, et al., “Dmdedup: Device mapper target for data deduplication,” in Proc. Ottawa Linux Symp., 2014, pp. 83–95.

[10] H. Yu, X. Zhang, W. Huang, and W. Zheng, “PDFS: Partially dedupped file system for primary workloads,” IEEE Trans. Parallel Distrib. Syst., vol. 28, no. 3, pp. 863–876, Mar. 2017.

[11] J. Kaiser, T. S€uß, L. Nagel, and A. Brinkmann, “Sorted deduplication: How to process thousands of backup streams,” in Proc. 32nd Symp. Mass Storage Syst. Technol., 2016, pp. 1–14.

[12] S. Jiang and X. Zhang, “LIRS: An efficient low inter-reference recency set replacement policy to improve buffer cache performance,” ACM SIGMETRICS Perform. Eval. Rev., vol. 30, no. 1, pp. 31–42, 2002.

[13] N. Megiddo and D. S. Modha, “ARC: A self-tuning, low overhead replacement cache,” in Proc. 2nd USENIX Conf. File Storage Technol., 2003, vol. 3, pp. 115–130.