International Journal of Computational Intelligence Research
ISSN 0973-1873 Volume 13, Number 10 (2017), pp. 2357-2363

http://www.ripublication.com

Alternative Approaches for Deduplication in Cloud Storage Environment

M. Tanooj Kumar* and M. Babu Reddy

Department of Computer Science, Krishna University,
Machilipatnam-521001, India
**Corresponding author*

Abstract

Cloud Storage is a model, in which data is stored on multiple dedicated online storage servers. Individuals, enterprises or organizations use private cloud storage to store their data. Since private cloud storage has limited resources, they need to be utilized optimally, by accommodating maximum data. Data deduplication is an effective data compression technique for eliminating duplicate copies of repeating data to optimize the utilization of storage space. This paper implements the data deduplication based on the file name through a case study and investigates its benefits and overheads.

Keywords: Cloud Storage, Private Cloud and Deduplication.

1. INTRODUCTION:

Cloud storage is a model, in which data is stored on multiple dedicated online storage servers [1]. Private cloud storage can be built by consolidating the storage resources of an organization. It has become an effective way for the organizations to store their massive data. However, with the increase of data quantity, communication traffic/ cost will increase. So its very important to seek an effective storage method to improve the utilization of storage space. The workload studies conducted by Microsoft [2, 3] and EMC [4, 5] suggest that about 50% and 85% of the data produced from primary and secondary storage systems, respectively, are redundant. The duplicated data may introduce more overhead of storage space and communications. Data deduplication is a technique for eliminating these redundant copies of data, used to improve storage utilization.

Data de duplication is a process in which only a single copy of the duplicate data is stored on the server. The metadata structures give information about the duplicates copies. This minimizes the storage space required and cost of maintaining storage space [6]. This paper presents a deduplicaiton system for private cloud storage, focuses on a specific case study involving a university scenario and studied the performance of that system. The rest of this paper is as per the following. Section-2 gives the background and motivation of the work, Section-3 explains the design of our deduplication framework, section-4 discusses the experimental setup and section-5 reports the results. Finally, concluding remarks were given in section-6.

2. BACKGROUND:

Data deduplication was proposed in 2000s to utilize the storage space optimally. In the deduplication process, duplicate data is identified and a single copy of the data is stored, along with references to the single copy of data by removing redundant data. The most common deduplication techniques are based on the fingerprints of files or chunks, computed using cryptographically secure hash algorithms. Duplicates are identified by matching their fingerprints.

Data deduplication systems can be classified into chunk level and file level deduplication systems. On the file level systems, each file will be hashed, and all these hash values are indexed. The drawback is that, if a part of a file is modified, no matter how small change in the content, treats the files are different and reduces the deduplication ratio. On the chunk level systems, data streams are divided into chunks, each chunk will be hashed, and all these hash values are indexed. The drawback is that, with the increase in quantity of hash values, it will occupy more RAM usage and increase lookup time. Based on these considerations, in this paper, a new data deduplication technique, called Blind Deduplication is proposed.

3. SYSTEM ARCHITECTURE:

The fundamental point of designing this system is to introduce a simple data deduplication technique for optimized cloud storage based on file names. The overall design of the system is shown in Figure-1. It is divided into three layers.

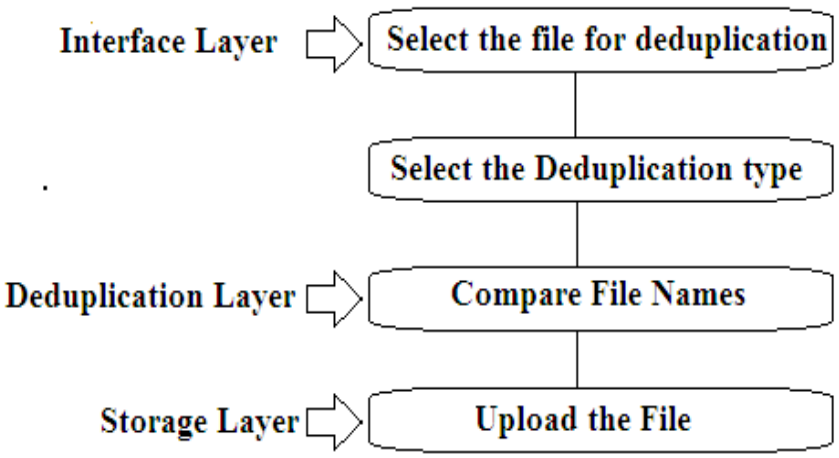

Figure-1: Overall Architecture

Interface Layer: First, the Interface layer gives the user interface to choose the file for deduplication and the type of deduplication.

Deduplication Layer: Deduplication layer detects the duplicate files, by using Blind deduplication algorithm, based on file names.

Blind Deduplication: Blind Deduplication is based on the information-rich naming procedure followed for file names. For example, University syllabus files, it is common to all the students studying in that university. If a single file is maintained in the cloud storage with a uniform name, all the students can access that file with out maintaining individual copies in that storage.

If any user desires to store a file, the user has to get approval for the file name for the first time. If the permissions are given, the user allowed storing the file on the cloud. For example, for the academic syllabus of CSE department, KU_R07_CSE_Syllabus.txt name is allocated. The size of the file is 475 KB .If every student maintains a copy; it needs 1GB memory for a single college. If this naming is followed and deduplication is used, it needs 2.75 MB memory. The advantage of this method is overhead computation of hash value calculation is not needed.

Algorithm 1: Blind Deduplication Algorithm

Input: File Selected

Output; File Uploaded to Cloud

Procedure: Blind Deduplication

begin Blind-Deduplication

/* select file to upload */

/* Read the input file name */

if (file exists)

{update index file}

else

{Store the file and update index file}

end if

end

Any content is added to the existing file; user can select file update option and mention the boundary values where the content is modified. The updated file can be saved with the file name with used-id and time stamp.

For Example, if the user wants to store the syllabus file with added references and notes, the user can store the updated file with the file name with used-id and time stamp.

Algorithm 2: Blind Deduplication Update Algorithm

Input: File Update Selected

Output; File Uploaded to Cloud

Procedure: Blind Deduplication Update

begin Blind-Deduplication-Update

/* select file to update */

/* Read the input file name */

if (file exists)

{newfile= strcat(filename,UserID,TimeStamp)}

end if

end

Storage Layer: After eliminating the duplicate files, new files are uploaded to cloud storage.

4. EXPERIMENTAL SETUP

Private Cloud storage is built with the open source software EUCALYPTUS (Elastic Utility Computing Architecture for Linking Your Programs To Useful Systems). Blind deduplication technique was implemented. A case study involving a university scenario has been studied and the corresponding results are discussed. A set of sample files (with and without duplicate copies) are taken for deduplication. If a new file comes, it is saved on the cloud storage. If a duplicated file comes, it is not saved, only index file is updated, the storage space is saved. Testing was done for different combinations of files and results were tabulated.

5. PERFORMANCE ANALYSIS

Table 1: TEST ITEM-1: Different text files with and without deduplication

S.No	File Name	File Size (KB)	No. Of Users	With out Deduplication (KB)	With Deduplication (KB)
1	Ku_R07_ece_syllabus.txt	416	120	49920	416
2	Ku_R07_eee_syllabus.txt	467	60	28020	467
3	Ku_R07_cse_syllabus.txt	480	120	57600	480
4	Ku_R07_it_syllabus.txt	480	60	28800	480
5	Ku_R07_me_syllabus.txt	475	120	57000	475
6	Ku_R07_ce_syllabus.txt	468	60	28080	468

Total Size (Before Deduplication) = 49920 + 28020 +57600 + 28800 + 57000 + 28080 = 249420 KB

Total Size (After Deduplication) = 416 + 467 + 480 + 480 + 475 + 468 = 2786 KB

Total Size Reduced = 249420 – 2786 = 246634 KB

Table 2: TEST ITEM-2: Updating the Existing text files with and without deduplication

S.No.	File Name	Original File Size With Deduplication (KB)	Updated file Size (KB)	Updated file With out Deduplication (KB)	Updated file With Deduplication (KB)
1	Ku_R07_cse_syllabus.txt	416	420	416	4

Total Size (Before Deduplication) = 420 KB

Total Size (After Deduplication) = 4 KB

Total Size Reduced = 420 – 4 = 416 KB

6. CONCLUSIONS

In conclusion, we have presented another cloud based data deduplication system. This system is mainly useful for the organizations and educational institutions, who desires to store their data over private clouds. This approach considers file name for deduplication to reduce computations of calculating hash values as in content aware deduplication and to manage mass data, it makes use of 'link files'. The experimental results show that a significant savings in the storage space of cloud is achieved.

REFERENCES

[1] Mikkilineni, D.R. and Sarathy, V. "Cloud Computing and the lessons from the past", IEEE international workshops on Enabling Technologies: Infrastructures for collaborative Entrprises, Los Altos, CA, 2009, pp. 4-5.

[2] Meyer, D. and Bolosky, W. "A study of practical deduplication", in proceedings of the USENIX conference on File and Storage Technologies (FAST'11). San Jose, CA, USA: USENIX Association, February 2011, pp. 229-241.

[3] El-Shimi, A. Kalach, R. Kumar A. et al., "Primary data deduplication-large scale study and system design", in proceedings of the 2012 conference on USENIX Annual Technical conference (USENIX'12). Boston, MA, SA: USENIX Association, June,2012, pp. 1-12.

[4] Wallace, G. Douglis, F. Qian, H. et al., "Characteristics of backup workloads in production systems", in proceedings of the Tenth USENIX conference on

file and storage technologies (FAST'12). San Jose, CA: USENIX Association, February 2012, pp. 1-14.

[5] Xia., W. jiang, H. Feng, D. et al., A comprehensive study of the past, present, and future of data deduplication, Proceedings of IEEE, 2016, 104 (9).

[6] Upadhya, A. "Deduplication and compression techniques in cloud design", IEEE International Systems Conference, Syscon 2012,