Cluster Based Cloud System For Effective Storage Capacity With The Help Of De-Duplication Technique

Archit Malpure[1], Gaurav Sonawane[2], Sudarshan Kapadnis[3], Rohit Khatwani[4], Prof. E. Jayanthi[5]

[1,2,3,4]*Student, Dept. of Computer Engineering, Sinhgad college of Engineering, Pune, Maharashtra, India.*
[5]*Assistant professor, Dept. of Computer Engineering, Sinhgad college of Engineering, Pune, Maharashtra, India.*

---***---

Abstract - *Data deduplication is one of the emerging technologies in storage right now. Users are generating lots of data right now and much of the data is duplicate even at the byte level. Deduplication technology can save lots of storage and in turn help in saving money. But current deduplication technologies are unable to deduplicate at the byte level. We wanted to Develop a cluster-based cloud system which will use its storage capacity effectively with the help of de-duplication technique. In simplified terms, data deduplication compares chunk of data and removes objects (copies) that already exist in the data set. The deduplication process removes blocks that are not unique. Once the data is chunked, an index can be created from the results, and the duplicates can be found and eliminated. Only a single instance of every chunk is stored. Every block or chunk of data stored on the disk is mathematically hashed and the hashes are indexed. Every new block of data being stored is also hashed, and the hashes are compared in the index. If the new data hash matches a hash for a block already stored, the new data does not get stored, thus eliminating duplicates. Our approach is to build in-line deduplication. Every uploaded file from client is break down in small chunks. The deduplication hash calculations are created on the target device as the data enters the device in real time. If the server doesn't find that similar block in its database, it will store this block meta-data in Db and save this chunk in disk cluster. If the Server spots a block that it already stored on the system it does not store the new block, just references to the existing block. Variable Chunking Will be done based on file type.*

***Key Words*: Data Deduplication, chunk, cluster, Hashing.**

1.INTRODUCTION

Data deduplication is an effective data compression technique which exploits data redundancy and partitions large data objects into smaller parts, called chunks. It represents these chunks with the help of digital fingerprints (cryptographic hash). Duplicate chunks are replaced by their fingerprints after chunk fingerprint lookup and only unique chunks are transferred to the storage. Data deduplication greatly increases the communication and storage efficiency by eliminating redundant chunks at storage.

Cloud computing is a service with potential to alter large part of IT industry. Cloud computing has centralized storage system for data. It also provides access to various computer services and resources. Maximizing the efficiency of shared resources is primary focus of cloud computing. World has crossed a billion Internet-connected computers. With widespread adoption of Internet- enabled smart phones, rise of cloud-based services and digitization of nearly every single photo, movie, song, and other file, data has grown exponentially, devices have proliferated and risk of data loss has increased by many folds.

Data deduplication can enable companies to save a lot of money on storage costs and on the bandwidth costs. If we can deduplicate what we store, we can better utilize our existing storage space, which can save a lot of money.

In simplified terms, data deduplication compares chunk of data and removes objects (copies) that already exist in the data set. The deduplication process removes blocks that are not unique.

2.LITERATURE SURVEY

The de-duplication identifies the duplicate data and eliminates it, by storing only one copy of the original data. If the duplicate occurs then the link will be added to the existing data. Also, paper tells us that, Data Deduplication describes approach that reduces the storage capacity needed to store data or the data has to be transfer on the network. Source Deduplication is useful in cloud backup that saves network bandwidth and reduces network space Deduplication is the process by breaking up an incoming stream into relatively large segments and deduplication each segment against only a few of the most similar previous segments. To identify similar segments, use block index technique. The problem is that these schemes traditionally require a full chunk index, which indexes every chunk, in order to determine which chunks have already been stored unfortunately, it is impractical to keep such an index in RAM and a disk-based index with one seek per incoming chunk is far too slow. In this paper we describe application-based deduplication approach and indexing scheme contains block that preserve caching which maintains the locality of the fingerprint of duplicate content to achieve high hit ratio and to overcome the lookup performance and reduced cost for cloud backup services and increase deduplication efficiency.

Bo Mao, Member, IEEE, Hong Jiang, IEEE Fellow, Suzhen Wu, Member, IEEE and Lei Tian, Senior Member, IEEE proposed a performance-oriented I/O deduplication, called POD, rather than a capacity- oriented I/O deduplication, exemplified by iDedup, to improve the I/O performance of primary storage systems in the Cloud without sacrificing capacity savings of the latter[1]. They have implemented a prototype of POD as a module in the Linux operating system. The experiments conducted on the lightweight prototype implementation of POD show that POD significantly outperforms iDedup in the I/O performance measure by up to 87.9% with an average of 58.8%. Moreover, their evaluation results also show that POD achieves comparable or better capacity savings than iDedup.

Yinjin Fu, Hong Jiang, Senior Member, IEEE, Nong Xiao, Member, IEEE, Lei Tian, Fang Liu, and Lei Xu present ALG-Dedupe, an Application-aware Local-Global source deduplication scheme that improves data deduplication efficiency by exploiting application awareness, and further combines local and global duplicate detection to strike a good balance between cloud storage capacity saving and deduplication time reduction[2]. They performed experiments via prototype implementation to demonstrate that the scheme can significantly improve deduplication efficiency over the state-of-the-art methods with low system overhead, resulting in shortened backup window, increased power efficiency and reduced cost for cloud backup services of personal storage.

According to research by Jin Li, Yan Kit Li, Xiaofeng Chen, Patrick P. C. Lee, Wenjing Lou convergent encryption technique has been proposed to encrypt the data before outsourcing[3]. To better protect data security, this paper makes the first attempt to formally address the problem of authorized data deduplication. Different from traditional deduplication systems, the differential privileges of users are further considered in duplicate check besides the data itself. They also present several new deduplication constructions supporting authorized duplicate check in a hybrid cloud architecture. Security analysis demonstrates that the scheme is secure in terms of the definitions specified in the proposed security model. As a proof of concept, they implement a prototype of their proposed authorized duplicate check scheme and conduct testbed experiments using prototype. They show that the proposed authorized duplicate check scheme incurs minimal overhead compared to normal operations.

3.PROPOSED SYSTEM

3.1 Fundamentals

Different data de-duplication products use different methods of breaking up the data into elements or chunks or blocks, but each product uses some technique to create a signature or identifier or fingerprint for each data element. As shown in the below figure, the data store contains the three unique data elements A, B, and C with a distinct signature. These data element signature values are compared to identify duplicate data. After the duplicate data is identified, one copy of each element is retained, pointers are created for the duplicate items, and the duplicate items are not stored. Based on the Technologies, or how it is done. Two methods frequently used for de-duplicating data are *hash based* and *content aware*.

Hash based Deduplication:

Hash based data de-duplication methods use a hashing algorithm to identify “chunks” of data. Commonly used algorithms are Secure Hash Algorithm 1 (SHA-1) and Message-Digest Algorithm 5 (MD5). When data is processed by a hashing algorithm, a hash is created that represents the data. A hash is a bit string (128 bits for MD5 and 160 bits for SHA-1) that represents the data processed. If you processed the same data through the hashing algorithm multiple times, the same hash is created each time.

Here are some examples of hash codes:

1. MD5 16-byte long hash:

# echo “The Quick Brown Fox Jumps Over the Lazy Dog” | md5sum 9d56076597de1aeb532727f7f681bcb0– # echo “The Quick Brown Fox Dumps Over the Lazy Dog” | md5sum 5800fccb352352308b02d442170b039d

Hash based de-duplication breaks data into “chunks”, either fixed or variable length, and processes the “chunk” with the hashing algorithm to create a hash. If the hash already exists, the data is deemed to be a duplicate and is not stored. If the hash does not exist, then the data is stored and the hash index is updated with the new hash.

2. Fixed-Length or Fixed Block:

In this data deduplication algorithm, it breaks the Data in to chunks or block, and the *block size or block boundaries* is Fixed like 4KB, or 8KB etc. And the block size never changes. While different devices/solutions may use different block sizes, the block size for a given device/solution using this method remains constant. The device/solution always calculates a fingerprint or signature on a fixed block and sees if there is a match. After a block is processed, it advances by exactly the same size and take another block and the process repeats.

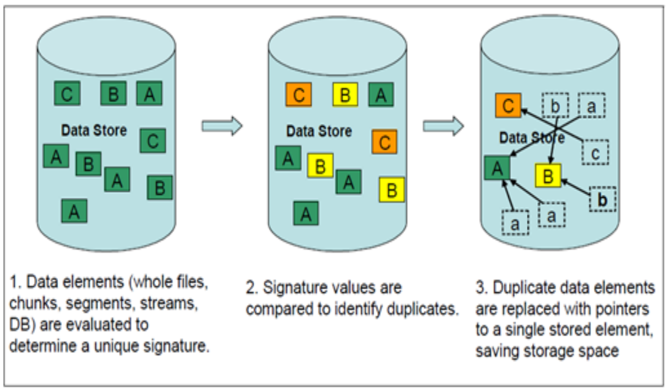

Data Store
Data Store
Data Store
1. Data elements (whole files, chunks, segments, streams, DB) are evaluated to determine a unique signature.
2. Signature values are compared to identify duplicates.
3. Duplicate data elements are replaced with pointers to a single stored element, saving storage space

Figure 1: Deduplication Process

3. Variable-Length or Variable Block:

In this data deduplication algorithm, it breaks the Data in to chunks or block, and the *block size or block boundaries* is variable like 4KB, or 8KB or 16KB etc. And the block size changes dynamically during the entire process. The device/solution always calculates a fingerprint or signature on a variable block size and sees if there is a match. After a block is processed, it advances by taking another block size and take another block and the process repeats.

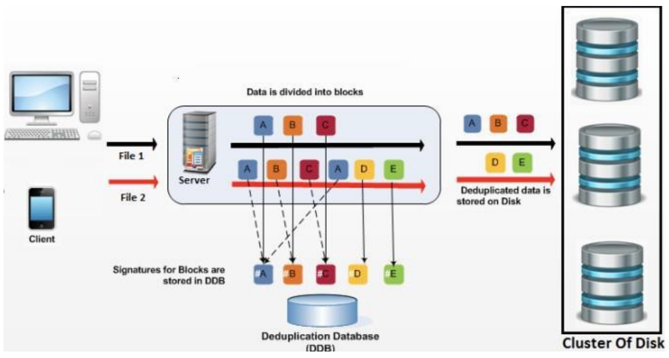

Data is divided into blocks
File 1
File 2
Server
Client
A
B
C
D
E
Deduplicated data is stored on Disk
Signatures for Blocks are stored in DDB
#A
#B
#C
#D
#E
Deduplication Database (DDB)
Cluster Of Disk

Figure 2: Architecture Diagram

3.2 System Overview

An architectural overview of-Deduplication is illustrated in, and data is broken into chunks by an intelligent chunker. Chunking is performed at the byte level with variable size for various type of files. Chunk size is set according to the file and the data is divided according to chunk size in bytes. Data chunks from the same type of files are then deduplicated in the deduplicator by generating hash fingerprints and performing data redundancy check. Their fingerprints are first looked up in an application-aware local index that is stored in the local disk for local redundancy check. If a match is found, the metadata for the file containing that chunk is updated to point to the location of the existing chunk. When there is no match, the fingerprint will be sent to the cloud for further parallel global duplication check on an application-aware global index, and then if a match is found in the cloud, the corresponding file metadata is updated for duplicate chunks, or else the chunk is new. Only those chunks which are not duplicate are stored in the database. Whenever a client wants to retrieve the data from the server the metadata of similar chunks are joined together and the corresponding chunks are sent to the client.

3.3 Algorithm Used:

Input: Uploaded file
Output: Download file
Steps:
1. Register a new user.
2. Login with newly created user details.
3. Upload a file.
4. Cut the file in chunks.
5. Upload chunk in database and update chunk info.
6. Calculate MD5 checksum of chunk and store in database.
7. If MD5 checksum is already present in database then just make a reference of that sum and don't store complete chunk in database.
8. Attempt to download that file.
9. System will gather all chunk info.
10. Validate MD5 checksum of each chunk.
11. Update user with this information.

4. METHODOLOGY

We obtained the results by performing deduplication on different datasets containing image files, text files, pdf files, audio files etc. It can be implemented on various storage systems such as disks, cloud systems etc. In the experiments, we also evaluate statistics of average response times for the read and write requests to evaluate the response time for various chunk sizes. We used machines having Android Operating System as a client which connected to the admin server and can be used to upload and download a file from the server. The data is encrypted during the communication phase using AES encryption to prevent loss of data.

The following modules are used for implementing the system: -

1. Web based GUI:

Server will be web-based application and this module will be responsible to take inputs from admin. The gui is developed in HTML and Java-script Our server input will be taken through this GUI where proper validations are supported. This includes new student registration, new assignment upload, etc.

2. Database Manager*:*

This module will help to handle all database related activity. All the SQL queries will be taken care in this module. A database connection polling system will be present to avoid repeatedly opening and closing database connection. The JDBC driver manager ensures that the correct driver is used to access each data source. The driver manager is capable of supporting multiple concurrent drivers connected to multiple heterogeneous databases.

3. MD5 checksum logic:

MD5 create the unique checksum of the given uploaded file of the user, if any user try to modify that file then system check current checksum with previous checksum if any data change is found then system detect the data leakage.

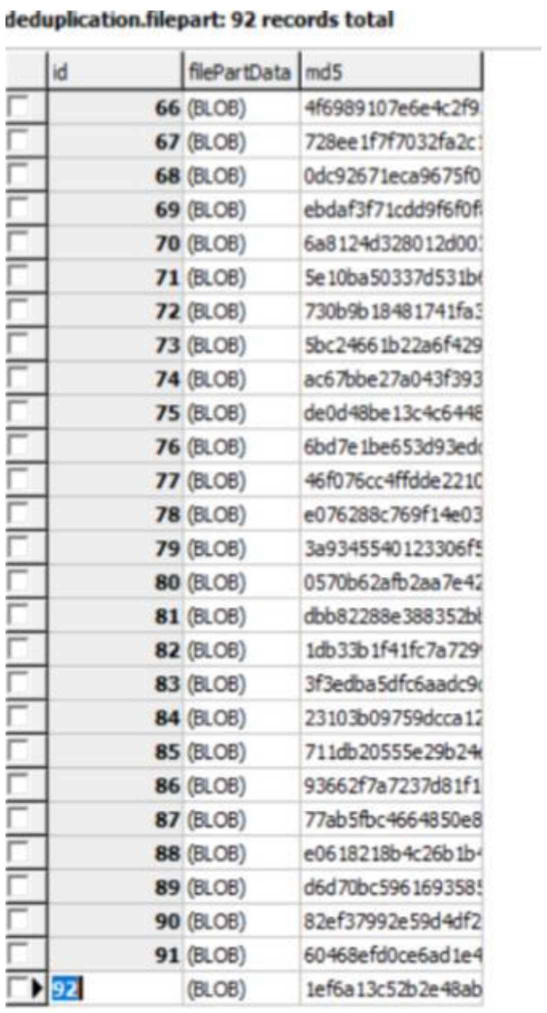
4. File Chunking:

This module takes the responsibility to create chunks of the updated files. It also maintains records of all chunks of uploaded files in database. System preferred the chunk size of 100 bytes.

5. Deduplication:

This module takes care of duplication of chunks every uploaded file is break down into small chunks of 100 bytes to 500 bytes as per file type .and these chunks are stored into database but before saving into database every chunks MD5checksum matched with previously uploaded local/global files

The deduplicated data can be seen in the database manager where the file chunks can be seen as file parts with each individual deduplicated chunk having a unique hash value.



deduplication.filepart: 92 records total

id	filePartData	md5
66	(BLOB)	4f6989107e6e4c2f9
67	(BLOB)	728ee1f7f7032fa2c
68	(BLOB)	0dc92671eca9675f0
69	(BLOB)	ebdaf3f71cdd9f6f0f
70	(BLOB)	6a8124d328012d00
71	(BLOB)	5e10ba50337d531b
72	(BLOB)	730b9b18481741fa
73	(BLOB)	5bc24661b22a6f429
74	(BLOB)	ac67bbe27a043f393
75	(BLOB)	de0d48be13c4c644
76	(BLOB)	6bd7e1be653d93ed
77	(BLOB)	46f076cc4ffdde221
78	(BLOB)	e076288c769f14e03
79	(BLOB)	3a93455401233306f
80	(BLOB)	0570b62afb2aa7e4
81	(BLOB)	dbb82288e388352b
82	(BLOB)	1db33b1f41fc7a729
83	(BLOB)	3f3edba5dfc6aadc9
84	(BLOB)	23103b09759dcca1
85	(BLOB)	711db20555e29b24
86	(BLOB)	93662f7a7237d81f1
87	(BLOB)	77ab5fbc4664850e8
88	(BLOB)	e0618218b4c26b1b
89	(BLOB)	d6d70bc596169358
90	(BLOB)	82ef37992e59d4df2
91	(BLOB)	60468efd0ce6ad1e4
92	(BLOB)	1ef6a13c52b2e48ab

Figure 3: Deduplicated Chunks with hash value

5.CONCLUSION

Thus, we are able to perform deduplication at the byte level which can help in deduplicating data very effectively and most importantly, save storage from duplicate byte data. We are able to perform this by creating chunks of data and assigning a hash value using a very reliable and secure MD5 hash function. It has a huge scope for reducing data load in the storage systems and help reduce network bandwidth. Data deduplication is a beneficial technology that can be applied to very large amount of data without compromising data integrity.

REFERENCES

[1] Mao, Bo, et al. ”Leveraging data deduplication to improve the performance of primary storage systems in the cloud.” IEEE Transactions on Computers65.6 (2016): 1775-1788.

[2] Yinjin Fu, Hong Jiang, Nong Xiao, Lei Tian, Fang Liu, and Lei Xu. “Application-Aware Local-Global Source Deduplication for Cloud Backup Services of Personal Storage” IEEE Transactions On Parallel and Distributed Systems (2014), vol. 25, no. 5

[3] Li, Jin, et al. ”A hybrid cloud approach for secure authorized deduplication.”IEEE Transactions on Parallel and Distributed Systems 26.5 (2015): 1206-1216.

[4] Xia, Wen, et al. ”A comprehensive study of the past, present, and future of data deduplication.” Proceedings of the IEEE 104.9 (2016): 1681-1710.

[5] Wen, Mi, et al. ”Secure Data Deduplication With Reliable Key Management for Dynamic Updates in CPSS.” IEEE Transactions on Computational Social Systems 2.4 (2015): 137-147.

[6] Zhou, Yongtao, et al. ” Reducing the read latency of in-line deduplication file system.” 2015 IEEE 34th International Performance Computing and Communications Conference (IPCCC). IEEE, 2015.