INTERNATIONAL JOURNAL OF RESEARCH
In Advanced Engineering Technologies

Volume 5, Issue 4 SEP 2016

IMPROVE THE PERFORMANCE OF BIG FILE CLOUD STORAGE WITH FILE COMPRESSION

[1] M. VASAVI, [2] J. NARASIMHA RAO

[1] M. Tech Student, Department of CSE, CMR Technical Campus, Village Kandlakoya, Mandal Medchal, District RangaReddy, Telangana, India.

[2] Associate Professor, Department of CSE, CMR Technical Campus, Village Kandlakoya, Mandal Medchal, District RangaReddy, Telangana, India.

Abstract— The use of Cloud-based storage services are rapidly increasing and becoming a trend in massive data storage fields. Cloud based storage is used by several users with massive storage capability for every user to store large amount of data. People use Cloud based Storage for backing up data, sharing the files to their friend. User stores large amount of file in Cloud and they may access that files later on. Due to massive amounts of data, system load becomes serious in cloud. There are several issues whereas accessing huge files like the processing of huge files, light-weight metadata and duplication etc. one of the solutions to resolve this drawback is light-weight data. The proposed System architecture i.e. BFC handled Big-Files based on Lightweight-metadata with file compression. Metadata for each file is created. Every file has the same size of metadata. The proposed System meets the user problem for handling Big-files in a Cloud and retrieval of Big-Files simply based on light-weight metadata.

Index Terms **Cloud Storage, Big File, Duplication of file, -- Lightweight metadata;**

I. INTRODUCTION

Traditional file systems has to face problem once managing a large number of huge File: how to balance system for the incredible growth of data to overcome this drawback, now a day's Cloud storage is widely utilized by people throughout the world within the form of cloud storage applications provided by cloud service providers. They provide the users the capability of storing the data within the form of files across many disks forming in a cloud. Cloud based mostly Storage services give large storage capacity wherever user will store large amount of data

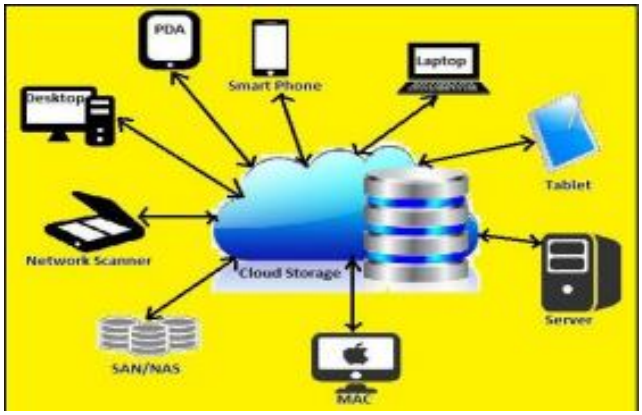


Figure 1: Cloud Storage System

People use cloud storage for their daily demands for e.g. data backup, sharing files to their friends via social networks like Google Drive, Zing Me etc. Users transfer great deal of data in Cloud using different types of devices like pc, laptop, and Mobile phone etc. They transfer or access that large quantity of data from Cloud soon. As a result of large amount of data, system load in Cloud is significant. To access large files easily and to ensure quality of service to the user, the systems are facing several issues. The users are expecting depth data

service for large number of users without bottleneck, Storing & Retrieving massive Files in System and managing them efficiently in system. System detects the data duplication to reduce the waste of storage space once user stores the same data. To overcome these issues, BFC i.e. the large Files storage using light-weight metadata with file compression is proposed here. The Big files are split into multiple smaller chunks; all chunks are encrypted then stored in cloud. Whereas downloading the file, chunks of that file get unified and the file is sent back to the user. The system detects the duplicate of files, creates for every chunk SHA value i.e. unique for each chunk. System detect duplicate of content of files and given them reference id.

II. RELATED WORK

The existing system authors projected Big-table; it is a distributed storage system for handling structured data. Big table is used to store very massive size of data which data is keep across thousands of artifact servers. Big table is employed by Google for many come. These applications have totally different demands of data size and latency needs. Big-table has provided high-performance solution for all Google products. Big-table provided the simple knowledge model that provides clients regarding data layout and format, and also the design and implementation of Big-table. Big-table is distributed storage system for storing structured data. The users just like the performance and high availability provided by the Big-table. In authors studied different techniques for storing and accessing Big-Files in Cloud and conjointly discussed the way to access Big-Files and how to remove duplication of same data to reduce storage space, network bandwidth, the encryption and decryption of data and replication of data for fault-tolerance and transmission of data in secure method for that purpose different protocols are used. In that authors provided Personal cloud storage services and that they provided for data-comprehensive applications. They provided a strategy to check capabilities and system design of private cloud storage services. They measured the implications of design choices on performance by analyzing different services. Their analysis shows the relevancy of client capabilities and protocol design to personal cloud storage services. Drop-box implements most of the analyzed capabilities, and its subtle consumer clearly enhancements performance, although some protocol possibly reduce network overhead. In authors provided Personal cloud storage services that are very popular. Cloud storage can quickly generate a large volume of net traffic because of huge number of providers offer service with low value for storage space. To handle increasing net traffic terribly limited is thought about the design and also the performance of systems, and the work of system. This understanding is important for designing cloud storage systems and predicting their impaction the network. They gave a characterization of Drop-box, the best leads to personal cloud storage. In authors implemented the Google filing system, an extensible distributed file system for applications. It implemented fault tolerance and it provides high performance to the quantity of clients. The file system has met storage needs successfully. It is used within Google because the storage system for the dealt of data utilized by analysis and service and also for development efforts that use large amount of data sets. The largest cluster provided very high storage space they can store large amount of data across variety of disks on over one thousand machines, and it is accessed by massive number of clients. They provided file system interface for distributed applications and sent measurements for micro benchmarks and real world use.

III. FRAME WORK

As shown in figure 2 there is cloud storage for user

where user can upload and download huge files of any type. While uploading the file, file is splitting into multiple chunks. The chunks are encrypted then stored in Cloud with compressed format and metadata is created for that specific file. At the user side it shows the number of chunks created on the cloud and also the size of that file. It additionally detects duplicate content of file if a similar file is uploading on the cloud for that file previous uploaded file reference is given to it file. Proposed System used MYSQL info for storing the user information and file information. File information includes information like user name, file name, fileid, SHA-value, reference id, start_chunk_id, num_chunks, file_size and status. To perform download operation the user choose the file name system merges all the chunks of specified file on cloud and sends file to the user. Steps to upload the File: a) the main function of System is splitting the large File into multiple smaller Chunks. b) Encode every chunk with AES-128 algorithm. c) System assigns the chunk id for those chunks. d) System finds the duplicates of files if any file information as user name, file name, fileid, Sha-value, reference id, start_chunk_id, num_chunks, file_size and status is stored in database. Steps to download the file: a) User requests the file to download from Cloud. b) System takes information from database and downloads the chunks of File name and prepares a file. System sends the requested file to the User.

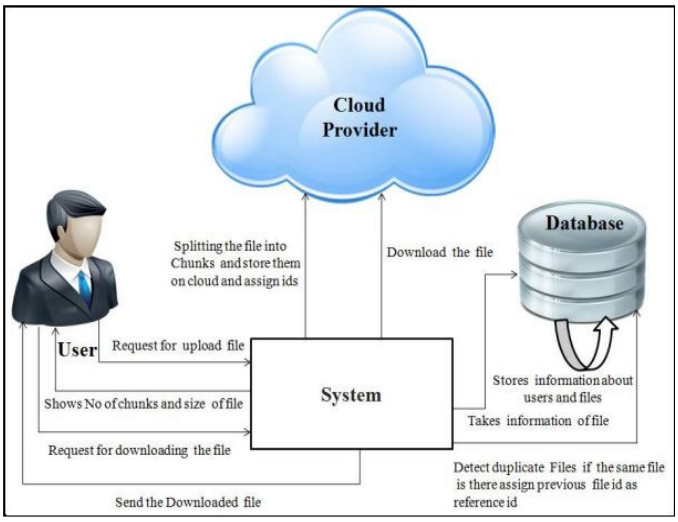


Figure 2: System Architecture

The different components that are implemented in system architecture are as follows:

A. Chunks Storage: In the cloud storage system the basic component is chunk. A chunk is a small section of data generated from a file. When a user uploads a file in the cloud, the size of the file is greater than 1MB, its split into multiple chunks. All generated chunks are of an equivalent size except the last chunk. System generates ids for all the chunks of the file. A File data object is made with data about the file like username, file name, file-id, SHA value, reference id, size of the file, id of the primary chunk, the number of chunks and status. The chunk is stored in key-value store with the key is the id of chunk and value is chunk data. FileInfo is consisting of following fields: filename - the file name, Fileid - unique identification of chunks of file, SHA1 - hash value by using sha-1 algorithm of file data, RefFileId - identification of file that have previous existed in System and have the same sha1 consider these files as one, StartChunkID - the identification of the first chunk of file, Num-Chunk- the quantity of chunks of the file File Size-size of file, status - the status of file.

B. Meta data Storage: In the planned system, the light-weight data for each uploaded file is created. The meta-data size is independent of number of chunks with any size of file. The size of metadata of file is same. The

metadata of file contains the file name, id of initial chunk, id of last chunk, and file size and SHA value i.e. unique code for every chunk. Meta-data is consisting of the following fields: filename - the file name, Fileid - unique identification of chunks of file, SHA1 - hash value by using sha1 algorithm of file data, Start ChunkID - the primary chunk id of file, FileSize - file size.

C. Duplication Mechanism: The planned System implemented duplication by using a simple technique with key-value store and SHA1 hash function to notice duplicate files within the system within the flow of uploading. A file content of various sizes is applied with SHA1 to get a key value and stored as SHA value. If a file with same text document with different file name is to be uploaded then same key are generated which can be the same as the previous key. During this situation file isn't uploaded on cloud so as to avoid the duplicity instead a reference id is copied from the id of the matched document.

D. Algorithm for upload of File:Step 1: System calculates SHA value of file contents. Step 2: System creates basic FileInfo like name, Fileid, Sha value, RefFileId, Start ChunkID, NumChunk, FileSize and standing, Step 3: send basic FileInfo to Cloud, Step 4: System Check whether SHA1 value Exists1. If exists then create FileInfo with refFileId, 2. If not server generates new FileId, new start ChunkID, create new FileInfo and send back to client.

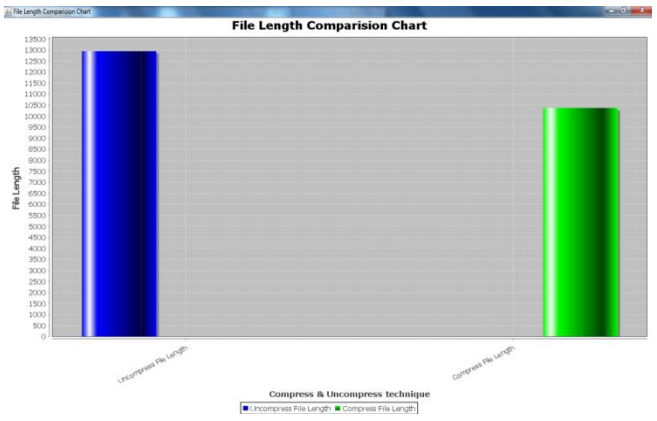
E. Algorithm for download of File: Step 1: System takes input filename, Step 2: System gets FileInfo from database, Step 3: System Prepare file based on FileInfo, fileSize, Step 4: download chunks from FileInfo, Starting ChunkID and fill them to prepare file.

F. Compression: In Cloud storage, various files may contain redundant data. This may result in vast amount of data being transmitted between user and also the server. One such solution which may overcome this drawback is to use further storage devices so as to enhance the current communication. However, this may lead to increased operation costs and time for the organization. One best resolution so as to overcome the problem of increased data storage and data transfer is by using efficient code. By implementing data compression, there's reduction in storage within the cloud thereby, enhancing the use of storage space. We can additionally say that the speed of data transfer will increase as compressed data is transferred between use and server systems.

IV. EXPERIMENTAL RESULTS

In our experiments, any number of users can registered and login into the system. Who are authorized users they can upload the files into the cloud. Those uploaded files are stored in chunk format in cloud. When we are trying to upload large files, in order to reduce space we first compress and store in the cloud. If any duplicate files are available in cloud, then that file cannot uploaded in the cloud and to that particular file reference will be assigned to the user. But that file can be downloaded by data owner as well as data users.

In the below chart we can observe that difference between the length of both compressed and uncompressed files.



We can observe that uncompressed file length is higher than compressed file length. The difference will be

shown in the sense of file length. So we can consider that the advantage of file compression.

Through our implementation we can store the big file in chunks format and detect the duplicate files as well as we can increase the storage space of cloud with file compression.

V.CONCLUSION

The proposed system presents architecture with objective to access easily the large files in the cloud using light-weight metadata. The size of metadata for every file is the same. It has been found that proposed system requires less uploading and downloading time as compared to BFC, Dropbox and normal technique. Also size of metadata is less as compared to other strategies. System also detects duplication of files using SHA value. Overall performance is improved using proposed system. We tested this system for different document and image files like doc, txt, pdf, jpeg, audio, video files like mp4 etc. we can store the big file in chunks format and detect the duplicate files as well as we can increase the storage space of cloud with file compression.

REFERENCES

[1] Thanh Trung Nguyen · Minh Hieu Nguyen, "Zing Database: High-Performance Key value Store for Large-Scale Storage Service", 17 August 2014, Springer - Vietnam J Comput Science (2015), DOI 10.1007/s40595-014-0027-4.

[2] Mihir Bellare, Sriram Keelveedhi, Thomas Ristenpart, "DupLESS: Server-Aided Encryption for Deduplicated Storage", 2013, USENIX Security Symposium.

[3] Fay Chang, Jeffrey Dean, Sanjay Ghemawat, Wilson C. Hsieh, Deborah A. Wallach, Mike Burrows, Tushar Chandra, Andrew Fikes, Robert E. Gruber, "Bigtable: A Distributed Storage System for Structured Data", Google, Inc.

[4] P. FIPS. 197: the official aes standard. "Figure2: Working scheme with four LFSRs and their IV generation LFSR1" LFSR, 2, 2001.

[5] S. Ghemawat and J. Dean."Leveldb is a fast key-value storage library written at google that provides an ordered mapping from string keys to string values."https://github.com/google/leveldb. Accessed November 2, 2014.

[6] S. Ghemawat, H. Gobioff, and S.-T. Leung. "The google file system". In ACM SIGOPS Operating Systems Review, volume 37, pages 29–43. ACM, 2003.

[7] Y. Gu and R. L. Grossman. "Udt: Udp-based data transfer for high-speed wide area networks." Computer Networks, 51(7):1777–1799, 2007.

[8] Martin Placek, Rajkumar Buyya, Taxonomy of Distributed Storage Systems". [9] T. Nguyen and M. Nguyen. Zing database: high-performance key-value store for large-scale storage service. Vietnam Journal of Computer Science, pages 1–11, 2014.

[9] P. ONeil, E. Cheng, D. Gawlick, and E. ONeil. The log-structured merge-tree (lsm-tree). Acta Informatica, 33(4):351–385, 1996.

[10] M. Placek and R. Buyya. A taxonomy of distributed storage systems. Reporte tecnico, Universidad de Melbourne, Laboratorio de sistemas distribuidosy computo grid, 2006.

[11] S. Di, D. Kondo, and W. Cirne, "Characterization and comparison of cloud versus grid workloads," in *International Conferenceon Cluster Computing (CLUSTER)*, 2012, pp. 230–238.

[12] .T.Nguyen and M.H.Nguyen, "Design Sequential Chunk identity with Light weight Metadata for Big File Cloud Storage", IJCSNS International Journal

of Computer Science and Network Security, VOL.15 No.9, September 2015.

[13] Jin Li, Xiaofeng Chen, Xinyi Huang, Shaohua Tang and Yang Xiang, Mohammad Mehedi Hassan, Abdulhameed Alelaiwi, "Secure Distributed De-duplication Systems with Improved Reliability", 2015, 10.1109/TC.2015.2401017, IEEE Transactions on Computers.

[14] Thanh Trung Nguyen · Minh Hieu Nguyen, "Zing Database: High-Performance Key value Store For Large-Scale Storage Service", 17 August 2014, Springer - Vietnam J Comput Sci (2015), DOI 10.1007/s40595-014-0027-4.

[15] J. Stanek, A. Sorniotti, E. Androulaki, and L. Kencl. A secure data de-duplication scheme for cloud storage. 2014.