ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 7, July 2016

BFCSULM: the System to Improve Performance of Big File Cloud storage using Lightweight Metadata

Supriya Survase, Manisha Nirgude

M.E Student, Dept. of C.S.E, Walchand Institute of Technology, Solapur, India

Assistant Professor, Dept. of I.T, Walchand Institute of Technology, Solapur, India

ABSTRACT: The use of Cloud-based storage services are rapidly increasing and becoming a trend in big data storage fields. Cloud based storage is used by many users with large storage capacity for each 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 large amounts of data, system load becomes heavy in cloud. There are many problems while accessing big files such as the Processing of Big files, lightweight metadata, duplication etc. One of the solutions to resolve this problem is Lightweight metadata. The Proposed System architecture i.e. BFCSULM handled Big-Files based on Lightweight-metadata. Metadata for every 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 easily based on Lightweight metadata.

KEYWORDS: Lightweight metadata, Big File, Cloud Storage, Duplication of file, Key-Value

I. INTRODUCTION

Traditional file systems has to face problem when managing a huge number of Big File: How to balance system for the incredible growth of data To overcome this problem, now a day's Cloud storage is widely used by people throughout the globe in the form of cloud storage applications provided by cloud service providers. They provide the users the capability of storing the information in the form of files across several disks forming a cloud. Cloud Based Storage services provide large storage capacity where user can store large amount of data. People use cloud storage for their daily demands for e.g. data backup, sharing files to their friends via social networks such as Google Drive, Zing Me etc. Users upload large amount of data in Cloud using different types of devices such as Computer, laptop, Mobile phone etc. They download or access that large amount of data from Cloud later on. Due to large amount of data, system load in Cloud is heavy. To access large files easily and to guarantee quality of service to the user, the systems are facing many problems. The users are expecting depth data service for large number of users without bottleneck, Storing & Retrieving Big Files in System and managing them efficiently in system. System detects the data duplication to reduce the waste of storage space when user stores the same data.

To overcome these problems, BFCSULM i.e. the Big Files storage using lightweight metadata is proposed here. The Big files are split into multiple smaller chunks, all chunks are encrypted and then stored in cloud. While downloading the file, chunks of that file get merged and the file is sent back to the user. The system detects the duplicate of files, creates for each chunk sha value i.e unique for every chunk. System detect duplicate of content of files and given them reference id.

II. RELATED WORK

In [1] authors proposed a system for stored large files. They designed a simple meta-data to create a high performance Cloud Storage based on Zing DB key-value store. They proposed approach in which they store key-value and used Zing Database (ZDB) which is a high-performance key-value store for improving reading and writing operations. ZDB uses powerful techniques to create key-value. To store a key-value pair in a file, they used hash function. They implemented Put, Get, and Remove operations in Zing Database.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 7, July 2016

In [2] authors proposed Bigtable, it is a distributed storage system for handling structured data. Bigtable is used to store very large size of data and that data is stored across thousands of commodity servers. Bigtable is used by Google for many projects. These applications have different demands of data size and latency requirements. Bigtable has provided high-performance solution for all Google products. Bigtable provided the simple data model which provides clients about data layout and format, and the design and implementation of Bigtable. Bigtable is distributed storage system for storing structured data. The users like the performance and high availability provided by the Bigtable.

In [3] authors studied different techniques for storing and accessing Big-Files in Cloud and also discussed how 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 way for that purpose different protocols are used.

In [4] authors provided Personal cloud storage services and they provided for data-comprehensive applications. They provided a methodology to check capabilities and system design of personal cloud storage services. They measured the implications of design choices on performance by analyzing different services. Their analysis shows the relevance of client capabilities and protocol design to personal cloud storage services. Dropbox implements most of the analyzed capabilities, and its sophisticated client clearly improvements performance, although some protocol possibly reduce network overhead.

In [5] authors provided Personal cloud storage services that are very popular. Cloud storage will quickly generate a large volume of Internet traffic because of huge number of providers provide service with low cost for storage space . To handle increasing internet traffic very limited is known about the architecture and the performance of systems, and the workload of system. This understanding is essential for designing cloud storage systems and predicting their impact on the network. They presented a characterization of Dropbox, the best results in personal cloud storage.

In [6] authors implemented the Google File System, an extensible distributed file system for applications. It implemented fault tolerance and it provides high performance to the number of clients. The file system has met storage needs successfully. It is used within Google as the storage system for the dealt of data used by research 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 number of disks on over a thousand machines, and it is accessed by large number of clients. They provided file system interface for distributed applications and sent measurements for micro-benchmarks and real world use.

In [7] authors developed protocols for large frequency in data transmissions. These are TCP protocol, which have determined better performance in simulation. Users who need to transfer bulk data they used application level solutions. The application levels protocols are UDP protocols, such as UDT protocol used for cloud computing. The major challenge for network designer's face is to achieve security of data and networks. Their earlier work analyzed various security methodologies which conduct to the development of a framework for UDT. They present less security by introducing an Identity Packet and Authentication Option for UDT. They introduced 'first packet identity' they created in such as way that receiver cannot be flooded by requests that require the receiver to take action before receiver have checked the identity and faith at the application level. They proposed security mechanism for UDT. They inspire the use of other hash functions, such as Secure Hash Algorithm-1 or Secure Hash Algorithm-256. They focused on the conceptual low-level protection of the end node. UDT depends on TCP and UDP protocol for data delivery. They proposed the inclusion of identity of receiver on its packet header (IP) and Authentication Option (AO) before the transmission is confirmed at the application level.

In [8] authors designed an encryption scheme that guarantees semantic security for unpopular data. They provide weaker security for popular data. They provide storage capacity and bandwidth for popular data. Data duplication can be powerful for popular data, while secure scheme protects unpopular content. This scheme is secure under the Symmetric External Decisional Diffie-Hellman and evaluated its performance with benchmarks and simulation and it scale for large number of users and files. In this system, encryption takes place at the client side and decryption is client-independent. File transmissions from one mode to other node takes place seamlessly at the storage server side if the file becomes popular.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 7, July 2016

III. PROPOSED ARCHITECTURE

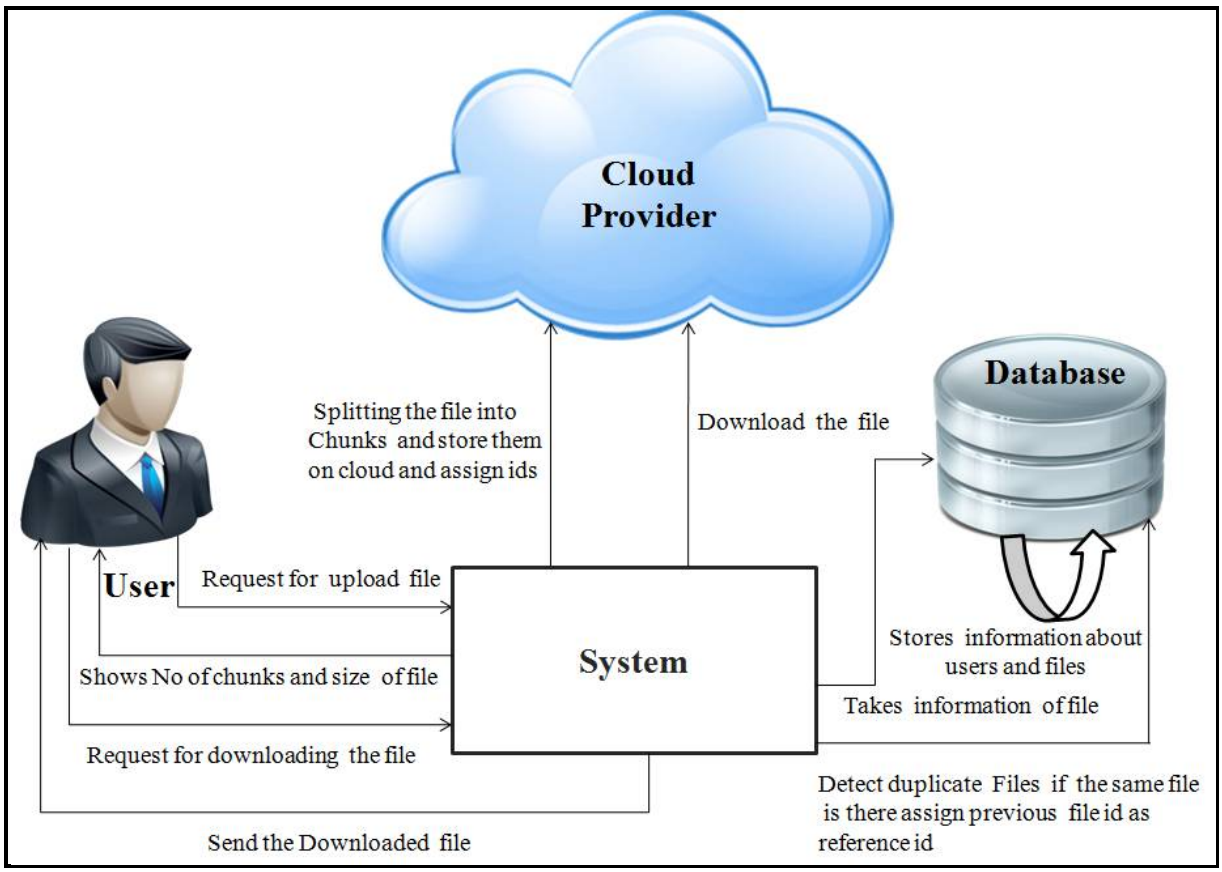

Fig1. System Architecture

As shown in Fig. 1, there is cloud storage for user where user can upload and download big files of any type. While uploading the file, file is splitted into multiple chunks. The chunks are encrypted and then stored in Cloud and metadata is created for that specific file. At the user side it shows the number of chunks created on the cloud and the size of that file. It also detects duplicate content of file if the same file is uploading on the cloud for that file previous uploaded file reference is given to that file.

Proposed System used MYSQL Database 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 select the file name system merges all the chunks of specified file on cloud and sends file to the user.

- Steps to Upload the File
  - The main function of System is splitting the Big File into multiple smaller Chunks
  - Encrypt each chunk with AES-128 algorithm
  - System assign the chunk id for that chunks
  - System finds the duplicates of files if any
  - File information as user name, file name, fileid, shavalue, reference id, start_chunk_id, num_chunks, file_size and status is stored in database
- Steps to Download the File
  - User request the file to download from Cloud
  - System takes information from database and downloads the chunks of File name and prepares a file.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 7, July 2016

- System send the requested file to the User

IV. IMPLEMENTATION

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

A. *Chunks Storage:*

In the cloud storage system the basic element 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 bigger than 1MB, it is split into multiple chunks. All generated chunks are of the same size except the last chunk. System generates ids for all the chunks of the file.

A File Info object is created with information about the file such as username, file name, file-id, sha value, reference id, size of the file, id of the first 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
- NumChunk- the number of chunks of the file
- FileSize-size of file
- Status - the status of file

B. *Meta Data Storage:*

In the proposed system, the lightweight metadata for every 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 first chunk, id of last chunk, and file size and sha value i.e. unique code for each 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
- StartChunkID - The first chunk id of file
- FileSize - file size

C. *Duplication Mechanism:*

The proposed System implemented duplication by using a simple method with key-value store and SHA1 hash function to detect duplicate files in the system in the flow of uploading.

A file content of various sizes is applied with SHA1 to generate 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 will be generated which will be similar to the previous key. In this scenario file is not uploaded on cloud in order 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 Filename, Fileid, sha value, RefFileId, StartChunkID, NumChunk, FileSize and status

Step 3: send basic FileInfo to Cloud

Step 4: System Check whether SHA1 Value Exists

1. If exists then Create FileInfo with refFileId
2. If not server generates new FileId, new startChunkID,create new FileInfo and send back to client

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 7, July 2016

1. upload chunks
2. Set Completed Status to FileInfo

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, StartChunkID and fill them to prepare file

V. RESULTS

In this section the performance analysis of the System with existing system i.e. dropbox is presented in [5] the performance is measured based on the time required for uploading the file and downloading the file which taking different types of files, images and audio etc. We used Amazon cloud for performance evaluation. The comparison of metadata of file for our system with existing system is shown based on size of metadata of file is considered.

A. *Upload Time of file*

Fig 2 shows the uploading time of different files is considered for the three systems as Normal, BFCSULM and Dropbox. From the figure it is clearly seen that uploading time required using the proposed system is less as compared to Dropbox and normal upload method.

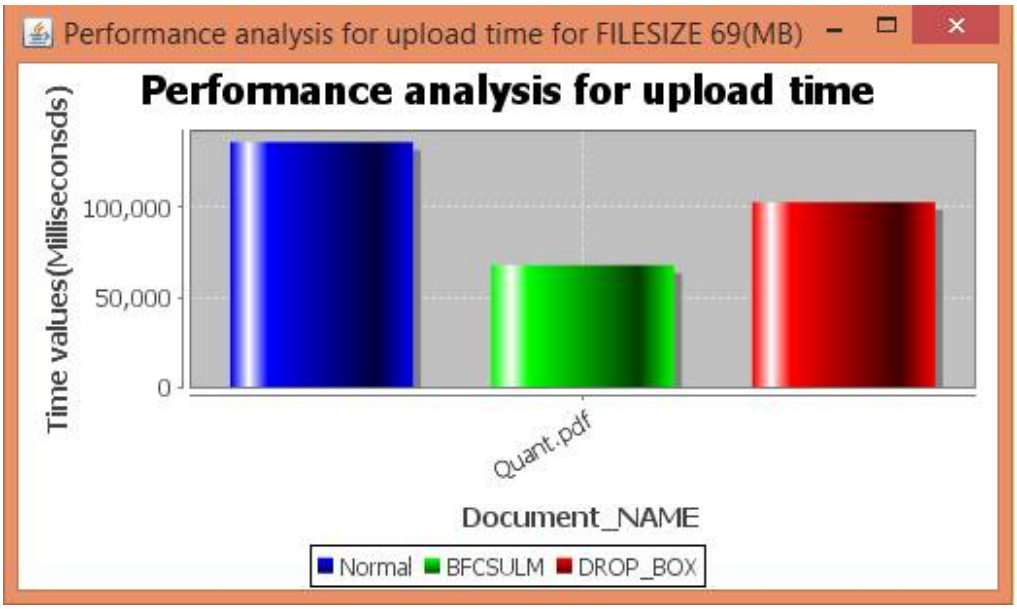

Fig.2. Comparison of upload time of file for different systems

B. *Download Time of file:*

Fig 3 shows the downloading time of different file is considered for three systems as Normal, BFCSULM and Dropbox. From the figure it is clearly seen that uploading and downloading time required using the proposed system is less as compared to Dropbox and normal download method.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 7, July 2016

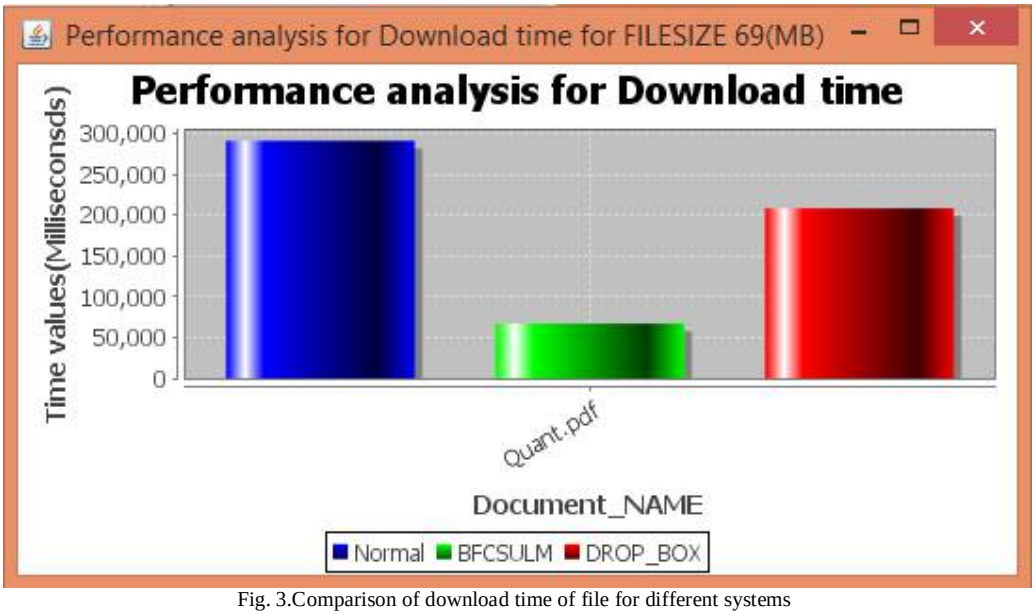

Fig. 3.Comparison of download time of file for different systems

C. *Metadata Comparison of File:*

Fig. 4 shows the metadata size of different files considered for two systems as BFCSULM and Dropbox. Metadata file size required using the proposed system is less as compared to Dropbox.

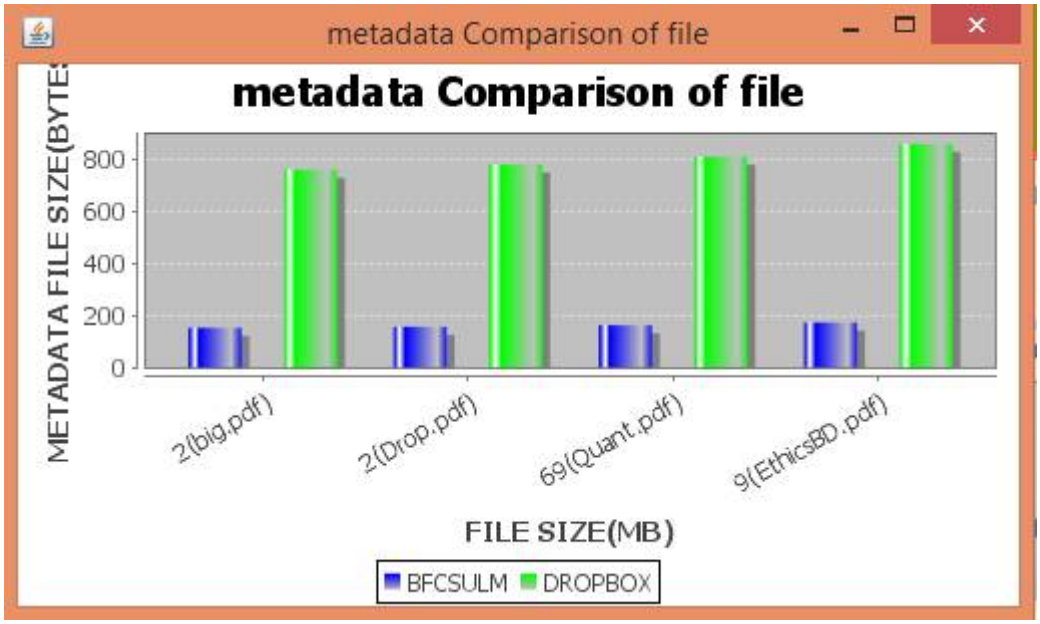

Fig. 4. Metadata comparison of different file sizes

Fig. 5 shows different metadata size of files considered for two systems as BFCSULM and Dropbox. Metadata file size required using the proposed system is less as compared to Dropbox.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 7, July 2016

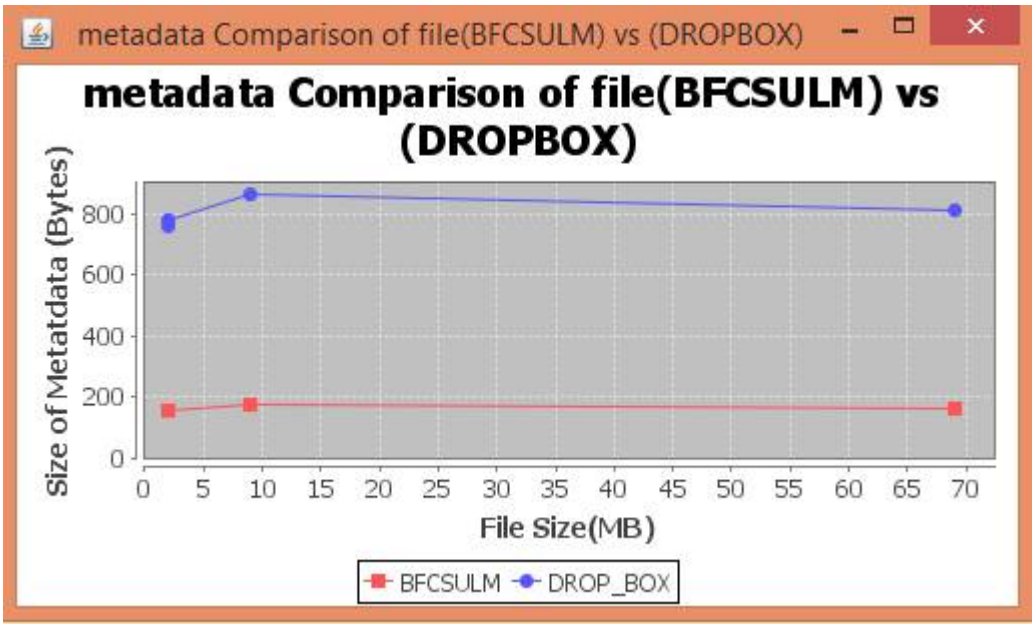

Fig. 5. Metadata Comparison of BFCSULM and Dropbox

D. *File Size Chart:*

Fig.6 shows the time taken for creation of chunks of file. From the figure it is clearly seen that the time taken for chunks creation of different files

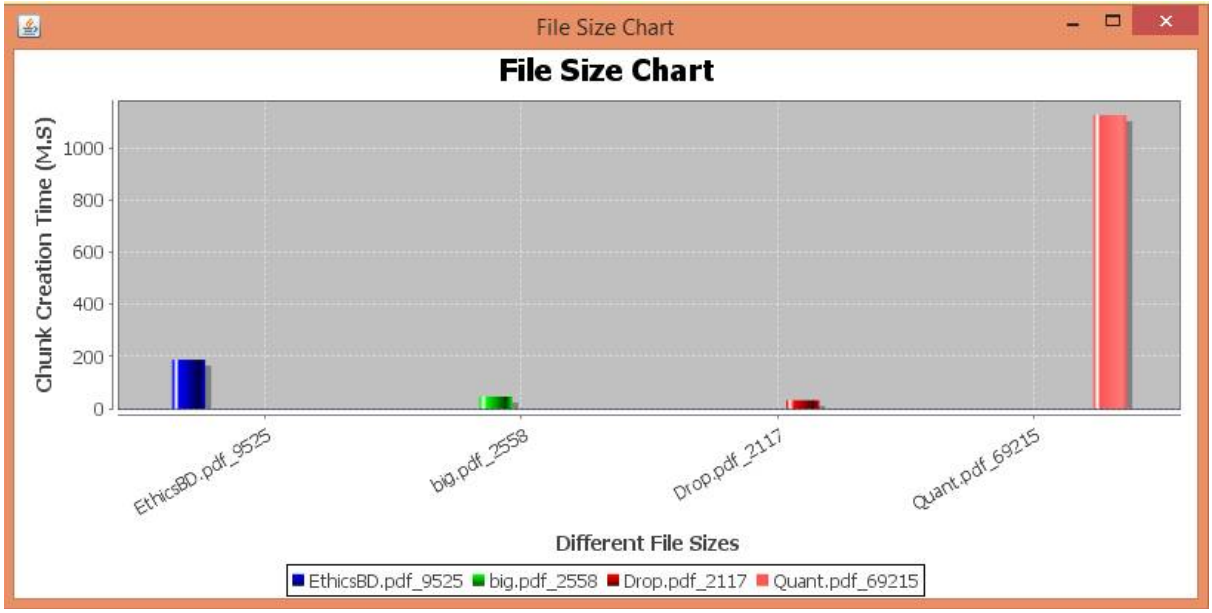

Fig 6. Comparison of time taken for creation of chunks for different files

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 7, July 2016

VI. CONCLUSION AND FUTURE WORK

The proposed system present architecture with objective to access easily the Big files in the cloud using lightweight metadata. The size of metadata for each file is the same. It has been found that proposed system requires less uploading and downloading time as compared to BFCSULM, Dropbox and normal method. Also size of metadata is less as compared to other methods. 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, pdf, jpeg, audio etc. In future, we will try for video files like mp4.

REFERENCES

1. Thanh Trung Nguyen, Tin Khac Vu, Minh Hieu Nguyen," BFC: High-Performance Distributed Big-File Cloud Storage Based On Key-Value Store", IEEE/ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD),vol. no., pp.1-6, 2015.
2. F. Chang, J. Dean, S. Ghemawat, W. C. Hsieh, D. A. Wallach, M. Burrows, T. Chandra, A. Fikes, and R. E. Gruber. "Bigtable: A distributed storage system for structured data", ACM Transactions on Computer Systems (TOCS), 2008.
3. Supriya Survase, Manisha Nirgude, "A Survey on Big-File Storing and Accessing in Cloud", In IJSRD - International Journal for Scientific Research & Development, Vol. 4, No. 02 , pp.2321-0613 ,2016.
4. I. Drago, E. Bocchi, M. Mellia, H. Slatman, and A. Pras, "Benchmarking personal cloud storage", ACM conference on Internet measurement conference in proceeding of the 2013,pp. 205–212, 2013.
5. I. Drago, M. Mellia, M. M Munafo, A. Sperotto, R. Sadre, and A. Pras, "Inside dropbox: understanding personal cloud storage services", ACM Conference on Internet measurement in Proceedings of the 2012, pp.481–494, 2012.
6. S. Ghemawat, H. Gobioff, and S.-T. Leung, "The google file system", ACM SIGOPS Operating Systems Review, Vol.37, No. 5,pp. 29–43, 2003.
7. Y.Gu and R. L. Grossman, "Udt: Udp-based data transfer for high-speed wide area networks", Computer Network, Vol. 51, No.7, pp. 1777–1799, 2007.
8. J. Stanek, A. Sorniotti, E. Androulaki, and L. Kencl, "A secure data deduplication scheme for cloud storage", 2014.

BIOGRAPHY

Supriya Survase obtained Bachelor of Engineering in 2014 from faculty of Information Technology, Walchand Institute of Technology Solapur University, Solapur. She is currently pursuing Master of Engineering from faculty of Computer Science and Engineering, Walchand Institute of Technology Solapur Maharashtra.

Manisha Nirgude obtained Bachelor of Engineering from Computer Science and Engineering and obtained Master of Engineering from faculty of Computer. She is Research Scholar and Assistant Professor at Walchand Institute of Technology, Solapur. She is pursuing doctoral study at Solapur University Solapur Maharashtra.