ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com

International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 4, Special Issue 16, April 2017

BFC: High-Performance Distributed Big-File Cloud Storage Based On Key-Value Store

[1] B. Raju, [2] M. Sreevani

[1]Asst. Professor, Department of CSE, Chaitanya Institute of Technology and Science, Kishanpura, Hanmakonda, Warangal, Telangana, India.

[2]M.Tech Student, Department of CSE, Chaitanya Institute of Technology and Science, Kishanpura, Hanmakonda, Warangal, Telangana, India.

Abstract— *Cloud storage provides a cost-effective and outstanding resolution to share the resources among cloud users. However, the storage capability within the cloud is increasing and has become a rising trend within the field of information storage. As the users load the information into the cloud there is also the possibilities of inflicting the main drawback like redundancy of files resulting in wastage of space for storing. To resolve this drawback Key-value store compete an important role and exhibited positive results. Knowledge with in the cloud has to be compelled to be supplied with security since the knowledge of the user is confidential . Here we tend to propose the idea of massive File Cloud with its evaluations and architectures for the massive storage of information stuffed by the users. A wants to upload a large file this large file is get compressed that compressed file is stored in cloud for reducing the storage space in the cloud. The researches additionally applied the benefits of Zing data Base (ZDB) which accurately resolved the problems in BFC. The outcomes are often used for building flexible circulated data distributed storage that backing vast record with size up to terabytes of information. Additionally here we tend to analyze and show the protection of the schemes.*

Keywords- Zing data base (ZDB), Key-value store, Big File cloud

I. INTRODUCTION

Storage pools are the adequate space assembled from various physical resources in a very distributed surroundings. In that cloud storage provides the space to stay the info for abundant users. For instance, capability for every user provided by the cloud infrastructure can be in gigabytes and terabytes. The recognition of the applications has become enormous since cloud is employed by the users for his or her lifestyle right from uploads, downloads of knowledge to exchange of data in any social networking sites, Zing me. So, the info drops in a very large repository are going to be large in cloud, increasing the holding ability of the system. Problems and difficulties are two-faced by the system to supply smart quality of services to the individuals. Thus these issues apply to the term.

The essential technique to tackle the difficulty is to cipher the data and afterward transfer constant. Miserably, providing an efficient and secured data-sharing theme is kind of difficult. On uploading files to big data by many varieties of users generally the documents could be similar and will cause to knowledge redundancy that is again a problem of waste within the space for storing. There are many different issues whereas designing an efficient storage engine similar to massive file process, de-duplication, distributed and high measurability.

Key value stores have various benefits for storing information in data-intensive operation. In recent years, key value stores have a really very good growth in each field. They have low latency with less time interval and high quantifiability with small and medium key value combine size. Current key value stores aren't designed for directly storing big-values, or big file in our case. We tend to executed many experiments within which we place whole file-data to key value store, the system failed to have sensible performance as usual for several reasons: first of all, the latency of put/get operation for big-values is high, thus it affects alternative parallel operations of key value store service and multiple concurrent accesses to totally different value. And, when the worth is massive, then there's no area to cache objects in memory for quick access. Finally, it's tough to scale-out system once number of users and information increase. This research is enforced to resolve those issues once storing big-values or big-file victimization key value stores. It has and gets several benefits of key value store in information management to analysis known as cloud-storage system referred to as Big File Cloud Storage (BFCS).

II. RELATED WORK

Maintaining the massive set of knowledge in correct manner becomes difficult. Knowledge confidentiality is additionally necessary when the files are distributed wide and is to be maintained with privacy protective needs. Transmission the data safely and accurately becomes difficult. Files sent to the server are divided into multiple range of chunks and stored it on a distributed system and are managed by data. But planning a light-weight data and storing these chunks and data with high performance were a number of the issues to be faced. The problem in storage space for data information resulted in O(n) and was unfeasible. a solution was found to use the advantages of key value store wherever the total file is place into it and the knowledge is hold on within the form of key and value combine as involving the info security, the files are encrypted and hold on within the storage area rather than keeping it directly.

Big File Cloud is introduced to manage the space for storing. For every divided file, every chunk is supplied with fixed size metadata; results in quick access distributed file IO. SHA256 algorithmic rule is applied and a key is generated to check for the redundancy of the content of the file meantime the comparison is created between the previous key and therefore the new key generated for future file and verify for the duplication of the file content. This paper proposes the benefits of ZDB within the sort of key value pair .An idea of digital signature is introduced to envision whether or not the user is authenticated whereas uploading the information to take care of the information privacy. a giant file table is maintained to stay track of the file details.

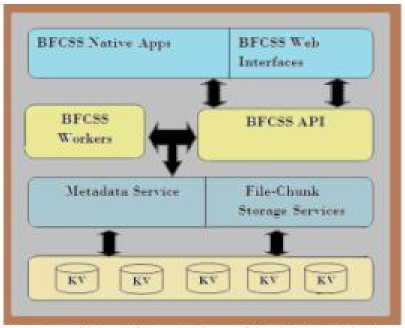


Figure 1: Shows the overview of BFCSS Architecture

For an enormous file a light-weight weighted metadata is planned, fixed size of data is given for each file. The space complexity drawback of data is solved that resulted in O (1), whereas the data size of a Drop box, HDFS has O(n) because the complexness, the initial file size is defined as n. To support serial scan and write operation ZDB is used, a little chunk-index overhead the file in the form of key value pair isn't sustained by utilizing its benefits. Information redundancy is reduced so the space for storing isn't wasted. Digital signatures are accustomed increase the authenticity. [7] discussed about a method, This scheme investigates a traffic-light-based intelligent routing strategy for the satellite network, which can adjust the pre-calculated route according to the real-time congestion status of the satellite constellation. In a satellite, a traffic light is deployed at each direction to indicate the congestion situation, and is set to a relevant color, by considering both the queue occupancy rate at a direction and the total queue occupancy rate of the next hop. The existing scheme uses TLR based routing mechanism based on two concepts are DVTR Dynamic Virtual Topology Routing (DVTR) and Virtual Node (VN). In DVTR, the system period is divided into a series of time intervals. On-off operations of ISLs are supposed to be performed only at the beginning of each interval and the whole topology keeps unchanged during each interval. But it has delay due to waiting stage at buffer. So, this method introduces an effective multi-hop scheduling routing scheme that considers the mobility of nodes which are clustered in one group is confined within a specified area, and multiple groups move uniformly across the network.

III. FRAME WORK

Usually when a file is uploaded a big file table is maintained which consists the following attributes.

- Chunk_id[i] = fileInform.firstChunk_id + I Meta-information is primarily represented in FileInform structure comprises of taking once fields:
- File_Name - the name of document;
- file_id: - one among a kind characteristic proof of record within the entire framework;

- SHA: - hash esteem by utilizing SHA calculation of document information;
- reference_file:- id of past existed go into Framework and have an equivalent sha256 – We regard these documents as one, reference_file is legitimate within the event that it's a lot of distinguished than zero;
- start_Chunkid : - the ID of the first lump of record, the subsequent piece can have id as start_Chunkid +1 et cetera;
- num_Chunk:- the number of chunks of the record;
- file_Size:- size of file in bytes;
- file_status:- the standing of record, it's one in four qualities to be specific

Status of the file could be:

Uploading File - once lump are transferring to server;

Completed File - once all blocks are transferred to server, and not checked as predictable;

Corrupted File - once all piece are transferred to server and not steady after checking;

Good Completed - the block of file is passed to the server and steady checking finished with result. By utilizing this arrangement, a light-weight meta-information is characterized in distributed storage.

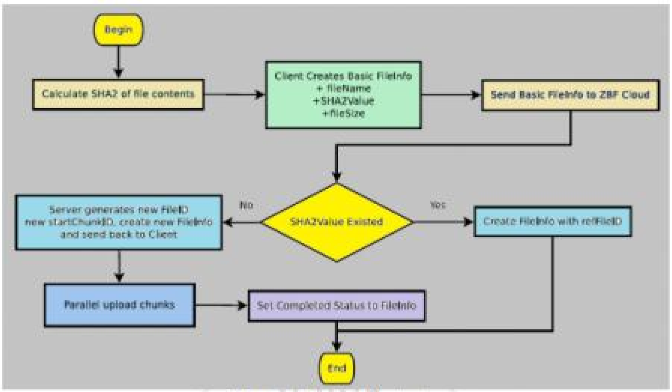

Begin
Calculate SHA2 of file contents
Client Creates Basic FileInfo
+ fileName
+SHA2Value
+fileSize
Send Basic FileInfo to ZBF Cloud
SHA2Value Existed
No
Yes
Server generates new FileID
new startChunkID, create new FileInfo
and send back to Client
Create FileInfo with refFileID
Parallel upload chunks
Set Completed Status to FileInfo
End

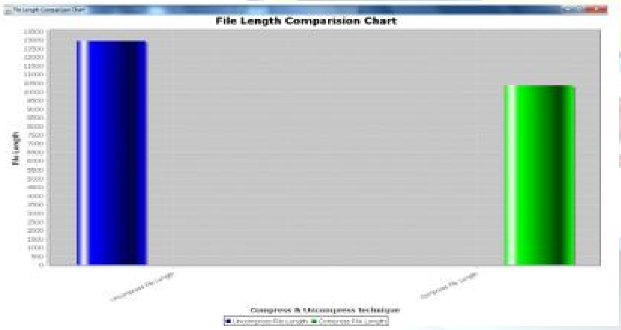
Figure 2: Uploading Mechanism

A text document of assorted size is applied with SHA256 to get a key value of 256bytes and keep as a file id and this file is get compressed and this compressed file is uploaded to the cloud. Because to reduce the storage space in the cloud. In case there's a second probability wherever a file with same text document with totally different file name is to be uploaded then same 256 bytes of key are going to be generated which is able to be the same as the previous key. This file isn't uploaded to the HDFS directory so as to avoid the duplicity .instead a reference id is derived from the id of the matched document. Group Signatures are used to accomplish secured information sharing. It enables the users to use the cloud anonymously. While user wants to download a file a copy of compressed file is downloaded. Because to save the bandwidth. This system also permits the users to transfer the information in an exceedingly secured manner. A pseudorandom numbers are generated by taking three variables and a signature is formed for the genuineness. Before the information is distributed to the cloud server the admin verifies the signatures and then the method is sustained.

For this there are two methodologies followed Signature Generation and Signature Verification. Once the information is uploaded to the HDFS directory it's stored in an encrypted format for further security purpose.

IV. EXPERIMENTAL RESULTS

In this application a large file uploaded to the cloud this file is gets compressed and that compressed file is stored in the cloud. Here the large file is split into number of small chunks. In that every chunk size is kept constant. If the other file with same file content is uploaded it is matched with the hash key and makes it as the reference Id. So to perform the de-duplication also. In that the size of the Metadata is linearly reduced. If the user wants to download the file that compressed file is downloaded because to save the bandwidth for downloading.



Here this graph is drawn for the file length and the file name. If you observed this graph compressed file take less length compared to uncompressed file (normal file) i.e. compressed file will occupy less storage space compared to the uncompressed file. So we finally conclude that if the file is stored in compressed format then easily save the storage space in the cloud.

V. CONCLUSION

In this paper designing massive File Cloud obtains high performance for distributed storage. Whenever the user uploads a file to the cloud typically identical redundant files may be stored that results in the waste in the storage capacity. To unravel the above methodology the paper uses a number of the algorithms like SHA256 that generates a key and is stored in a very massive file table once uploading a file. The info duplication is checked on the server side. If the opposite file with same file content is uploaded it's matched with the hash key and build it because the reference Id. The files area unit divided into chunks and supplied with IDs. The benefit of key value store into massive files of knowledge is used. Sequential scan and write operation is completed by ZDB. Confidentiality of data is maintained by preserving the file information by providing an authentication of the user exploitation Digital Signatures. The info might be shared simply privacy protected without revealing any identity privacy. The safety is ensured and certifies the potency further.

REFERENCES

[1] Thanh Trung Nguyen, Tin Khac Vu, Minh Hieu Nguyen, Ha Noi, Viet Nam, “BFCSS: High-Performance Distributed Big-File Cloud Storage Based On Key value Store”, June 1-3 2015, IEEE SNPD 2015, 978-1- 4799-8676-7/15(Base Paper).

[2] T.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.

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

[4] D. Borthakur."Hdfs architecture guide."HADOOP APACHEPROJECThttp://hadoop.apache.org/common/docs/current/hdfs design. Pdf, 2008.

[5] I. Drago, M. Mellia, M. M Munafo, A. Sperotto, R. Sadre, and A. Pras."Inside dropbox: understanding personal cloud storage services". InProceedings of the 2012 ACM conference on Internet measurement conference, pages 481–494. ACM, 2012

[6] S. A. Weil, S. A. Brandt, E. L. Miller, D. D. E. Long, and C. Maltzahn. "Ceph: A scalable, high performance distributed file system". In Proceedings of the 7th Symposium on Operating Systems Design and Implementation, OSDI '06, pages 307–320, Berkeley, CA, USA, 2006. USENIX Association.

[7] Christo Ananth , P.Ebenezer Benjamin, S.Abishek, "Traffic Light Based Intelligent Routing Strategy for Satellite Network", International Journal of Advanced Research in Biology, Ecology, Science and Technology (IJARBEST), Volume 1,Special Issue 2 - November 2015, pp.24-27.

[8] F. Chang, J. Dean, S. Ghemawat, W. C. Hsieh, D. A. Wallach, M. Burrows, T. Chandra, A. Fikes, and R. E. Gruber. "Bigtable: Adistributed storage system for structured data". ACM Transactions on Computer Systems (TOCS), 26(2):4, 2008.

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

[10] 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 stringvalues." https://github.com/google/leveldb. Accessed November 2, 2014.

[11] 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.

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