Indian J.Sci.Res. 17(2): 539-542, 2018

ISSN: 0976-2876 (Print)
ISSN: 2250-0138(Online)

IMPLEMENTING FILE COMPRESSION AND IMPROVING HIGH PERFORMANCE FOR BIG FILE CLOUD

[1]K. Krishna Priyanka
[1]Department of Computer Science and Engineering, Aurora's Scientific, Technologicaland Research Academy, Bandlaguda, Hyderabad

Abstract-: Cloud services are provides services to the consumers. Cloud based storage services are rapidly growing and make a new trend in data storage field. In the previous days there are many problems when designing an efficient storage engine for cloud-based systems with some requirements such as processing of big file, lightweight meta-data, low latency, parallel compression of I/O data, sharing, high Capability of a system growth some amount of data. Key-value stores played an important role and give many advantages when solving those problems. This paper presents about Big File Cloud (BFC) with its value store. low-complicated, fixed-size meta-data design, which supports fast and operating at the same time and running parallel, sharing of file I/O, more algorithms for fit re-sumable upload, download and simple data deduplication method for static data.

Keywords: Cloud Services, Social Network, Load Balancing, Cloud Based System.

I. Introduction

Cloud-based storage services commonly serves millions of users with storage capacity for each user can reach to several gigabytes to terabytes of data. People use cloud storage for the daily demands, for example backing-up data, sharing file to their friends via social networks such as Face book, Zing Me. Users also probably upload data from many different types of devices such as computer, mobile phone or tablet. After that, they can download or share them to others. System load in cloud storage is usually really heavy.

Serving intensity data service for a large number of users without bottle-neck; Storing, retrieving and managing big-files in the system efficiently; Parallel and resumable uploading and downloading; Data deduplication to reduce the waste of storage space caused by storing the same static data from different users. In traditional file systems, there are many challenges for service builder when managing a huge number of big file: How to scale system for the incredible growth of data;How to distribute data in a large number of nodes; How to replicate data for load-balancing and fault-tolerance; How to cache frequently accessed data for fast I/O, etc. A common method for solving these problems which is used in many Distributed File Systems and Cloud Storages is splitting big file to multiple smaller chunks, storing them on disks or distributed nodes and then managing them using a meta-data system . Storing chunks and meta-data efficiently and designing a lightweight meta-data are significant problems that cloud storage providers have to face. After a long time of investigating, we realized that current cloud storage services have a complex meta-data system; at least the size of metadata is linear to the file size for every file. Therefore, the space complexity of these meta-data system is $O(n)$ and it is not well scalable for big-file. In this research, we propose new big-file cloud storage architecture and a better solution to reduce the space complexity of meta-data.

Key-Value stores have many advantages for storing data in data-intensity services. They often outperform traditional relational databases in the ability of heavy load and large-scale systems. In recent years, key-value stores have an unprecedented growth in both academic and industrial field. They have low-latency response time and good scalability with small and medium key-value pair size. Current key-value stores are not designed for directly storing big-values, or big file in our case. We executed several experiments in which we put whole file-data to key-value store, the system did not have good performance as usual for many reasons: firstly, the latency of put/get operation for big-values is high, thus it affects other concurrent operations of key-value store service and multiple parallel accesses to different value reach limited. Secondly, when the value is big, there is no more space to cache another objects in main memory for fast access operations. Finally, it is difficult to scale-out system when number of users and data increase. This research is implemented to solve those problems when storing big-values or big-file using key-value stores. It brings many advantages of key-value store in data management to design a cloud-storage system called Big File Cloud (BFC). These are our contributions in this research: – Propose a light-weight meta-data design for big file. very file has nearly the same size of meta-data. BFC has $O(1)$ space complexity of meta-data of a file, while size of meta-data of a file in Dropbox[1], HDFS[4] has space complexity of $O(n)$ where n is size of original file. See Fig

[1]**Corresponding Author**

9 – Propose a logical contiguous chunk-id of chunk collection of files. Those make it easier to distribute data and scale-out the storage system. – Bring the advantages of key-value store into big-file data store which is not default supported for big-value.ZDB[16] is used for supporting sequential write, small memory-index overhead. These contributions are implemented and evaluated in Big File Cloud (BFC) that serve storage for Zing Me Users. Disk Image files of VNG's CSM Boot diskless system are stored in Big File Cloud.

II. Related Work

Cloud-based capacity administrations are quickly developing and turning into a rising pattern in information stockpiling field. There are numerous issues when planning a proficient stockpiling motor for cloud-based frameworks with a few prerequisites, for example, enormous document handling, lightweight meta-information, low inactivity, parallel I/O, deduplication, circulated, high adaptability. Key-worth stores assumed an essential part and indicated numerous points of interest when taking care of those issues. This paper presents about Big File Cloud (BFC) with its calculations and construction modeling to handle most of issues in a major document distributed storage framework in view of key value store. It is finished by proposing low-confused, settled size meta-information outline, which backings quick and exceedingly simultaneous, dispersed record I/O, a few calculations for resumable transfer, download and basic information deduplication technique for static information. This examination connected the upsides of ZDB - an in-house key value store which was upgraded with auto-increase whole number keys for taking care of enormous document stockpiling issues proficiently. The outcomes can be utilized for building versatile appropriated information distributed storage that bolster huge document with size up to a few terabytes.

III. Frame Work

A common method for solving these problems which is used in many Distributed File Systems and Cloud Storages is splitting big file to multiple smaller chunks, storing them on disks or distributed nodes and then managing them using a meta-data system. Storing chunks and meta-data efficiently and designing a lightweight meta-data are significant problems that cloud storage providers have to face. After a long time of investigating, we realized that current cloud storage services have a complex meta-data system; at least the size of metadata is linear to the file size for every file. we propose new big-file cloud storage architecture and a better solution to reduce the space complexity of meta-data.

A. Detection of service

A common method for solving these problems which is used in many Distributed File Systems and Cloud Storages is splitting big file to multiple smaller chunks, storing them on disks or distributed nodes and then managing them using a meta-data system. Storing chunks and meta-data efficiently and designing a lightweight meta-data are significant problems that cloud storage providers have to face. After a long time of investigating, we realized that current cloud storage services have a complex meta-data system; at least the size of metadata is linear to the file size for every file. low-complicated, fixed-size meta-data design, which supports fast and operating at the same time and running parallel, sharing of file I/O, more algorithms for fit resumable upload, download and simple data deduplication method for static data. This research applied the advantages of key value store which is make best or more effective use with auto-increment integer keys for solving big-file storage problems efficiently. Finally the results can be used to make able to change the size in distributed data cloud storage that will support big-file with size up to a millions of information.

B. Overall Description

A Software Requirements Specification (SRS) – a requirements specification for a software system is a complete description of the behavior of a system to be developed. It includes a set of use cases that describe all the interactions the users will have with the software. In addition to use cases, the SRS also contains non-functional requirements. Nonfunctional requirements are requirements which impose constraints on the design or implementation (such as performance engineering requirements, quality standards, or design constraints).

System requirements specification: A structured collection of information that embodies the requirements of a system. A business analyst, sometimes titled system analyst, is responsible for analyzing the business needs of their clients and stakeholders to help identify business problems and propose solutions. Within the systems development lifecycle domain, the BA typically performs a liaison function between the business side of an enterprise and the information technology department or external service providers. Projects are subject to three sorts of requirements:

Preliminary investigation examine project feasibility, the likelihood the system will be useful to the organization. The main objective of the feasibility study is to test the Technical, Operational and Economical feasibility for adding new modules and debugging old running system. All system is feasible if they are unlimited resources and infinite time. There are aspects in the feasibility study portion of the preliminary investigation

Application Layer: It consists of native software on desktop computers, mobile devices and web-interface, which allow user to upload, download and share their own files.

Storage Logical Layer: it consisted of many queuing services and worker services, ID-Generator services and all logical API for Cloud Storage System. This layer implements business logic part in BFC.

Object Store Layer: It contains many distributed backend services. Two important services of Object Store Layer are FileInfoService and ChunkStoreService. FileInfoService stores information of files. Y-value store mapping data from fileID to FileInfo structure.ChunkStoreService stores data chunks which are created by splitting from the original files that user uploaded.

Persistent Layer: it based on ZDB key-value store. There are many ZDB instances which are deployed as a distributed service and can be scaled when data growing.

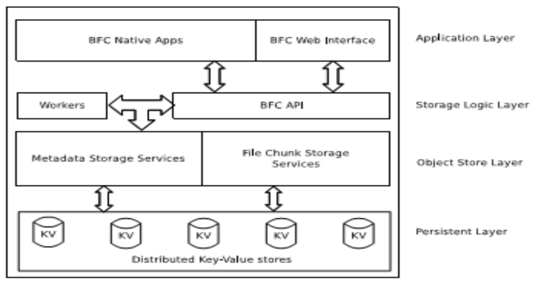

Fig. 1. BFC Architecture

Fig: 1. BFC Architecture

Fig.1. Cloud services are provides services to the consumers. Cloud based storage services are rapidly growing and make a new trend in data storage field. In the previous days there are many problems when designing an efficient storage engine for cloud-based systems with some requirements such as processing of big file, lightweight meta-data, low latency, parallel compression of I/O data, sharing, high Capability of a system growth some amount of data. Key-value stores played an important role and give many advantages when solving those problems. This paper presents about Big File Cloud (BFC) with its valuestorewhich supports fast and operating at the same time and running parallel, sharing of file I/O, more algorithms for fit resumable upload, download and simple data deduplication method for static data. This research applied the advantages of key value store which is make best or more effective use with auto-increment integer keys for solving big-file storage problems efficiently. Finally the results can be used to make able to change the size in distributed data cloud storage that will support big-file with size up to a millions of information.

IV. Experimental Results

Our implementation and experiments were developed to validate and examine the overall performance of each the credibility model and the provision model.After compress the files we can do the more compression in large memory files into zip files.

Fig:2 File information on screen

Fig: 3 Compression Chart

V. Conclusion

Cloud services are provides services to the consumers. Cloud based storage services are rapidly growing and make a new trend in data storage field. In the previous days there are many problems when designing an efficient storage engine for cloud-based systems with some requirements such as processing of big file, lightweight meta-data, low latency, parallel compression of I/O data, sharing, high Capability of a system growth some amount of data. Key-value stores played an important role and give many advantages when solving those problems. This paper presents about Big File Cloud (BFC) with its value store. low-complicated, fixed-size meta-data design, which supports fast and operating at the same time and running parallel, sharing of file I/O, more algorithms for fit resumable upload, download and simple data deduplication method for static data.

References

[1] D. Borthakur. Hdfs architecture guide.HADOOP APACHE PROJECT http://hadoop.apache.org/common/docs/current/hdfs design. pdf, 2008

[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), 26(2):4, 2008.

[3] L. Chappell and G. Combs. Wireshark network analysis: the official Wireshark certified network analyst study guide. Protocol Analysis Institute, Chappell University, 2010.

[4] I. Drago, E. Bocchi, M. Mellia, H. Slatman, and A. Pras. Benchmarking personal cloud storage. In Proceedings of the 2013 conference on Internet measurement conference, pages 205–212. ACM, 2013.

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

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

[7] 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.

[8] 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.

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

[10] P. Hunt, M. Konar, F. P. Junqueira, and B. Reed. Zookeeper: wait-free coordination for internet-scale systems. In Proceedings of the 2010 USENIX conference on USENIX annual technical conference, volume 8, pages 11–11, 2010.