A Survey on Big-File Storing and Accessing in Cloud

Supriya Survase[1] Manisha Nirgude[2]

[1,2]Walchand Institute of Technology Solapur

Abstract— Cloud-based storage services are rapidly increasing and becoming an emerging trend in data storage fields. Cloud based storage are commonly used by millions of users with large storage capacity for each user to store large amount of data. People use Cloud Based Storage for their daily demands for backing up data, sharing the files to their friend via Social Network. Users stores large amount of data in Cloud and they may access that large amount of data later on. Due to large amount of data, system load is heavy in cloud. There are some problems in processing Big-files in Cloud. In this survey paper, we discussed different methods which dealt with the problem of accessing Big-Files easily in cloud. One of the solutions to resolve problem is Lightweight metadata i.e. Key-Value store in Database.

Key words: Big File, Cloud Storage, Distributed Storage, Lightweight Metadata, Key-Value

I. Introduction

Traditional file systems has to face many problems for service builder when managing a huge number of Big File: How to balance system for the incredible growth of data; How to distribute data in a number of nodes; How to replicate data in multiple nodes for load-balancing and fault-tolerance; To overcome such a problems, now a days Cloud-based storage services are commonly used by many users. Cloud Based Storage is a model of data storage where user stores large amount of data. Cloud Based Storage Services Servers for millions of user with large storage capacity for each user can reach to several gigabytes to terabytes of data. People uses 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, Facebook etc. Users upload large amount of data in Cloud using different types of devices such as Computer, laptop, Mobile phone etc. Later on, they download or access that large amount of data from Cloud. 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 shall consider issues like data deduplication to reduce the waste of storage space when user stores the same data, parallel uploading and downloading, Data distribution and replication for fault tolerance and load balancing.

Key-Value stores have many advantages for storing the data in data-intensity services. Key-value stores have enormous growth in storage field. Key-Value have low-latency response time and good scalability with small and medium key-value pair size.

II. Literature review

In this section, we discussed different techniques for storing and accessing Big-Files in Cloud .Some problems and their solution on Cloud are listed below:

F. Chang, J. Dean, S. Ghemawat proposed Bigtable,[1] it is a distributed storage system for handling structured data. Bigtable is designed to store very large size of petabytes 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 on Bigtable, in terms of data size and latency requirements. Bigtable has provided high-performance solution for all Google products. They described the simple data model which gives clients dynamic control over data layout and format, and also the design and implementation of Bigtable. They have described Bigtable is distributed storage system for storing structured data. The users like the performance and high availability provided by the Bigtable and they can increase the capacity of clusters by simply adding more machines into the system as the resource demands change over time.

I. Drago, E. Bocchi, M. Mellia provided Personal cloud storage services they are used for data-comprehensive applications producing a significant share of Internet traffic.[2] Different companies offered several solutions for attracting more and more people. However, very little is known about service capabilities, architecture of system and performance of design choices. They presented a methodology to study cloud storage services. They apply their methodology to compare 5 different trendy offers, revealing different system architectures and capabilities. The implications on performance of different designs are checked executing a series of benchmarks. Their results show no clear winner, with all services having some limitations or having some improvement. In some situations, the upload of the same file can take times more, wasting twice as much capacity. Their methodology and results are useful as benchmark and guideline for system design. In this paper they presented a methodology to check both 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.

I. Drago, M. Mellia, M. M Munafo, A. Sperotto studied on Personal cloud storage services they are very popular.[3] With a rush of providers to provide services enter the market and an increasing offer of low-cost storage space, cloud storage will quickly generate a high amount of Internet traffic. 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. They analyzed data from four vantage points in Europe, collected during 42 consecutive days. They provide 3 contributions are First, they are the first to study Dropbox, they showed the most widely-used cloud storage system, accounting for a volume equivalent to one third of the YouTube traffic at campus networks. Second, they characterized the workload users in different environments generate to the system and highlighting how this workload reflects on network traffic. Last, their results show possible performance bottlenecks caused by the current system architecture and the storage protocol. This is for users connected far from storage data-centers.

S. Ghemawat, H. Gobioff have designed and implemented the Google File System, [4] a scalable distributed file system for applications. It implemented fault tolerance and it provides high performance to a large number of clients. System design has been driven by examination of their application workloads and technology, both present and foreseen that reflect a marked from some earlier file system assumptions. The file system has successfully met storage needs. It is deployed within Google as the storage system for the processing of data used by service as well as research and development efforts that use large amount of data sets. The largest cluster provided very high storage space they can reach from hundreds of terabytes of storage of data across large number of disks on over a thousand machines, and it is accessed by hundreds of clients. They presented file system interface for distributed applications and report measurements for micro-benchmarks and real world use. The Google File System determines the qualities necessary for supporting large-scale data processing workloads on inexpensive commodity hardware. Some design decisions are different many may apply to data processing tasks of a similar consequence and cost consciousness. They started by re-examining traditional file system assumptions in light of their present and foreseen application workloads and technological environment.

P. Hunt, M. Konar described Zookeeper,[5] a service for coordinating processes of distributed applications. Zookeeper is part of critical framework, Zookeeper provided high performance kernel for building more complex coordination primitives at the client side. It merge elements from group messaging, shared registers, and distributed lock services. The interface of zookeeper has the wait-free aspects of shared registers with an event-driven mechanism to provide a simple, powerful coordination service. The Zookeeper interface set up a high-performance service implementation.

P. Jin, P. Yang, and L. Yue proposed a new B+-tree-based index for hybrid storage systems,[6] which is called Hybrid B+tree. The Hybrid B+ tree aims to reduce the random writes to SSD and keeping high time performance and low buffer costs. They introduced huge leaf to avoid the splits and merges of data on B+tree. A vast leaf node consist of two or more leaf nodes in different states. They place the leaf nodes on HDD or SSD and according to their current states, and dynamically maintain the states of leaf nodes where they are read or updated. They described the structure and operations of the Hybrid B+tree, they give analysis on the costs of the Hybrid B+tree. Then, they conduct experiments on two TPC-C, using a real hybrid storage system. Hybrid storage system includes one Hard Disk Drive and two Solid State Drive, and then compares the performance of their proposal solution with two B+-tree implementations, they are B+-tree on HDD and the B+-tree on SSD/HDD. The results show the best time performance and the fewest buffer costs.

D. Karger, A. Sherman studied on Performance measure for the World Wide Web is the speed with which content is provided to users.[7] As the traffic on the Web increases, users are faced problems with increasing delays and failures in data delivery. Web caching is one of the approach to improve performance. An important issue in many caching systems is how to determine what is cached where at any given time. Solutions to overcome this issue are multicast queries and directory schemes. They described a new Web caching strategy based on consistent hashing. Consistent hashing provides multicast and directory schemes, and has several advantages in load balancing and fault tolerance. They described a consistent-hashing-based system implementation and it can provide performance improvements.

Y.Gu and R. L. Grossman studied on the emergence of various new technologies has pushed researchers to develop new protocols that support high frequency data transmissions in WAN.[8] Many of these protocols are TCP protocol, which have determine better performance in simulation and have several limited network experiments but they have limited practical applications because of installation and implementation of system difficulties. Users who need to transfer bulk data they used application level solutions. Protocols used in the application level are UDP-based protocols, such as UDT used for cloud computing. The major challenge for network designer's face is to achieve security of data and networks. Their previous 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 also introduced and proposed security mechanism for UDT and in future implements the various network topologies. They demonstrated the use of MD5, while they encourage 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.

R. van Renesse and F. B. Schneider proposed Chain replication for coordinating clusters of storage servers.[9] This approach is designed for supporting large-scale storage services that show high throughput and availability without sacrificing strong consistency guarantees. Likewise outlining the chain replication protocols themselves, the simulation experiments of chain replication explore the characteristics of performance of a prototype implementation. In this way they discuss Throughput, Availability and Object Placement Strategy. When chain replication is occupied, high availability of data objects comes from when carefully selecting a strategy for placement of volume of data replicas on servers.

J. Stanek, A. Sorniotti, E. Androulaki designed an encryption scheme that guarantees semantic security for unpopular data. [10]They provide weaker security for popular data. They provide storage capacity and bandwidth for popular data. In this way, data deduplication can be powerful for popular data, while secure encryption 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 numbers 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 that file becomes popular.

III. Conclusion

Cloud-based storage services are commonly used by many users for different applications with high storage capacity. It is a daily demand of people to share, upload and download big files using different devices. Users stores large amount of data on cloud and they may access that large amount of data later on. Due to large amount of data, System load is heavy in Cloud. In this survey paper, we discussed how to access Big-Files and how to remove deduplication of same data to reduce storage space, network bandwidth and replication of data for fault-tolerance. We are proposing big-file cloud storage system based on Light weight metadata i.e. key value store in Database. We will create Light weight metadata for big files. Light weight metadata will be easy stored and it will be helpful to improve the performance.

Acknowledgments

I would like to thank Department of Computer Science and Engineering, Walchand Institute of Technology Solapur.

References

[1] 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.

[2] 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.

[3] 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.

[4] 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.

[5] 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.

[6] P. Jin, P. Yang, and L. Yue. “Optimizing b+-tree for hybrid storage systems”, Distributed and Parallel Databases, pages 1–27, 2014.

[7] D. Karger, A. Sherman, A. Berkheimer, B. Bogstad, R. Dhanidina, K. Iwamoto, B. Kim, L. Matkins, and Y. Yerushalmi. “Web caching with consistent hashing”, Computer Networks, 31(11):1203–1213, 1999.

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

[9] R. van Renesse and F. B. Schneider. “Chain replication for supporting high throughput and availability”, In OSDI, volume 4, pages 91–104, 2004.

[10] J. Stanek, A. Sorniotti, E. Androulaki, and L. Kencl. “A secure data deduplication scheme for cloud storage”, 2014.