# Analysis of key technologies of distributed file system based on big data

ZHOU JUNPING[1]

**Abstract.** With the advent of the big data era, it is becoming more and more urgent to provide a stable and efficient distributed file system. The performance of traditional distributed file systems is becoming increasingly difficult to meet the growing needs of the Internet in terms of scalability, stability, and access to multiple users. The key technologies of big data oriented distributed system - Clover cluster system were designed in this paper. In this system, the two-stage mapping relation and two segment decision protocol were adopted to ensure the operability, extensibility and availability of metadata. The performance test shows that the Clover cluster has good operability, scalability and availability in metadata, and can be used as the key technology for big data oriented distributed systems.

**Key words.** Big data analysis, distributed file systems, Clover cluster.

## 1. Introduction

With the development of information technology, there have been more and more kinds of information in data center and Internet, and semi-structured and unstructured data has become explosive increase in geometric quantity [1]. These are signs of the era of big data. Compared with the previous data, the information in the big data era presents the following characteristics:

(1) With the rapid growth of massive data files, the size of the file has expanded rapidly. In large-scale Internet Co, such as Alibaba, Tencent and so on, the size of their data has already exceeded the PB level, and the amount and size of documents to be processed is beyond measure [2].

(2) User access: for some large Internet Co in China, it is common to say that more than one million or ten million people live online at the same time. The large amount of random read and write data caused by so many users accessing online at the same time is fatal to the entire system, whether it is storage or access [3].

(3) Data structures and processing are varied. With the steady development of

[1]Ulanqab Vocational College, Wulanchabu City in Inner Mongolia, 012000, China

Internet applications, the types of Internet applications have become multifarious [4]. For both offline data transmission and online data analysis, the system needs to provide uninterrupted high-quality services, so that the system security requirements are getting higher and higher [5].

With the advent of the big data era, the demand for storage systems has become even severer. The storage system should not only support data processing and analysis of data center and Internet applications, but also should store a large amount of data in a timely and effective way, which is embodied in the following aspects:

(1) In the face of large amounts of data and huge files, the storage of the system needs better scalability and concurrent processing capability.

(2) The system can support different kinds of access requirements in different ways.

(3) The sharing of data and the security of data are becoming more and more important. Under the premise of providing full freedom and sharing, the system should also ensure the safety and reliability of data [6].

## 2. State of the art

The distributed file system refers to the physical storage resources that are managed by the local file system and are connected to the nodes through the Internet network rather than directly connected to the local nodes [7]. The design of a distributed file system comes from the client server model. In a full network, the server can provide services that many users access online simultaneously [8]. At the same time, the network also allows some systems to act as both users and servers [9]. For example, a user releases data that other users can also access, and when other users access the data, the data is local drivers for the client.

## 3. Methodology

Specifically, all kinds of distributed file systems can adapt to a certain application environment and play a superior performance and well meet the needs of the computing system for storage system at the various stages of storage technology development [10]. The design and implementation of the distributed file system are based on a certain storage structure, which is organized according to the storage medium, and the storage structure adopted in the distributed file system is mainly divided into virtualization storage structure and object storage structure [11]. The more successful distributed file storage systems currently used are GFS, HDFS, PNUTS, etc.

- Research on GFS architecture: GFS supports massive data block storage. For file updates, GFS is done by adding new data rather than modifying the raw data [12]. In the GFS architecture, the stream mode of a large number of data is the read operation, and the random mean of a small amount of data is the read operation [13]. As can be seen from the above, GFS performs better in large-scale search services, but it also limits GFS's business applications in other areas.

- Research on HDFS architecture: HDFS supports user or user programs to create folders and stores data and documents [14]. Users can use the permissions to create, delete, move, and rename the folder, but HDFS does not support user disk quotas and access restrictions. In data storage, HDFS also supports data block storage, while the disk data errors, heartbeat detection and re-replication are features of HDFS [15].

## *3.1. Clover file system*

The Clover distributed file system uses the multiple data server architecture, which can be completely compatible with the HDFS interface by separating the data stream and the control flow. While ensuring the functionality of the file system, it also enables the use and scalability of the very high meta data, and the operation of metadata in the no data server can also be unified.

A typical Clover cluster is shown in Figure 1, the underlying layer consists of several data servers (datanode, DN), and metadata servers are linked (metadata, server, MDA), a cluster is formed by storing, analyzing, and storing data and metadata with a shared storage pool, and the services are provided to clients primarily with client nodes.

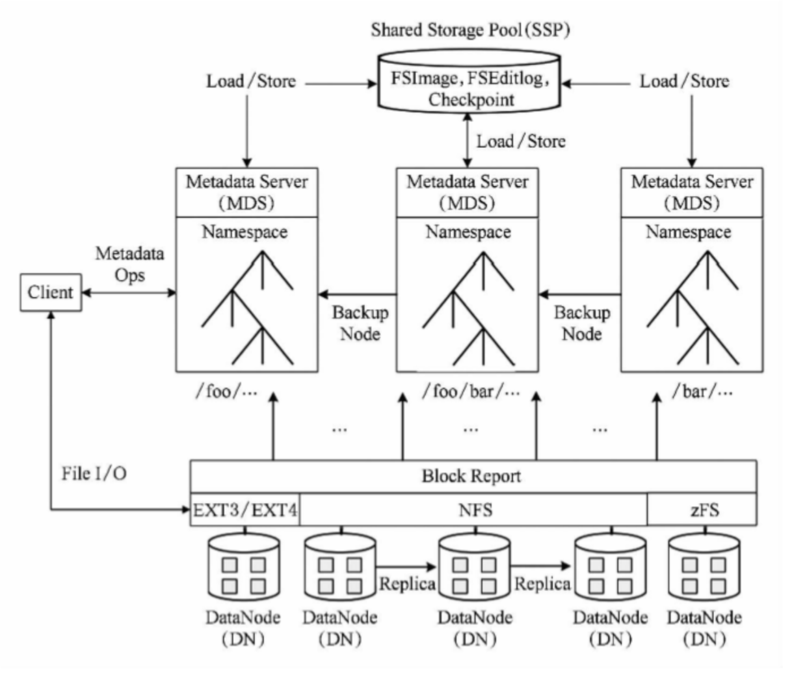

Fig. 1. Typical structure of Clover cluster

In the Clover cluster, the files are stored on a specified number of data servers in block form, and the data servers are placed in duplicate copies of the blocks of files, and the data is checked on the data server. When the metadata server sends a command to the file, the relevant data server sends the file of the genus to the corresponding node according to the block information, so as to reduce the memory occupancy of metadata servers and optimizing the system. At the same time, similar to the HDFS disk data error and heartbeat detection function, the Clover cluster also periodically detects nodes on the metadata server and uploads the test files corresponding to the information and content claims to the shared storage pool. The node detection of the metadata server can be directed to a single metadata server node, or to all metadata server nodes within the cluster, at the same time, the corresponding nodes can be backed up and saved, so that when the fault occurs, the backup can be replaced quickly, and the functions of the cluster can be quickly restored.

When the metadata server issues a request to process metadata, the corresponding metadata server will form a corresponding cluster, then store metadata related information, and provide corresponding functions, such as file retrieval, file attributes, file names, and so on. At the same time, the cluster stores the corresponding contents written in the shared storage pool, so that the next operation can be done quickly and effectively. While writing content to a shared storage pool, the metadata server also writes the content back to the local server, and thus the reliability and convenience of the next operation are guaranteed by comparing the two contents in the shared memory and the local server.

## *3.2. Extensibility of metadata*

By reasonably dividing and mapping the spatial information of the names, the division of namespaces can be self-adaptable and extended, and can adapt to the rapidly growing amount of data and files, so as to save space and optimize the system. In the Clover cluster, the Hash is used to partition namespaces.

When data is stored into a number of metadata servers, the operations such as renaming, copying, pasting can affect more than just a metadata server, which can cause a large amount of metadata to be transmitted in the metadata server, and make system pressure. Through the global distributed directory Hash table (global, Hash, table, GDT), the mapping relationship between directory and metadata server is established. In this mapping relationship, there are two levels in the Clover cluster. The mapping of the first level is the storage path name of the file or data, and the Hash is calculated to find the GDT mapping relation. The second level is based on the first level, and the mapping relation of one to one is obtained. Through the consistency Hash algorithm, the nearest metadata server node is searched in the Clover cluster and mapped to the corresponding metadata server.

GDT likes a huge combo cabinet, each metadata service is a drawer on the cabinet with different maps. GDT is composed of these mapping tables, and each mapping table is stored on different metadata servers in blocks. Figure 3 shows the process of metadata manipulation by using the GDT principle. The dotted line represents the corresponding files generated during the process, but when the operation is completed, the system is deleted. Starting with the mapping of the first level, the GDT on the metadata server 2 where the /bin/foo7 is stored is searched. Then, by using the second level algorithm, the relevant metadata information is directly operated on the metadata server 2. The advantage of the process is to operate metadata directly on the metadata server, so as to avoid space occupation caused by metadata migration, greatly save the space of the system and optimize the structure of the system.

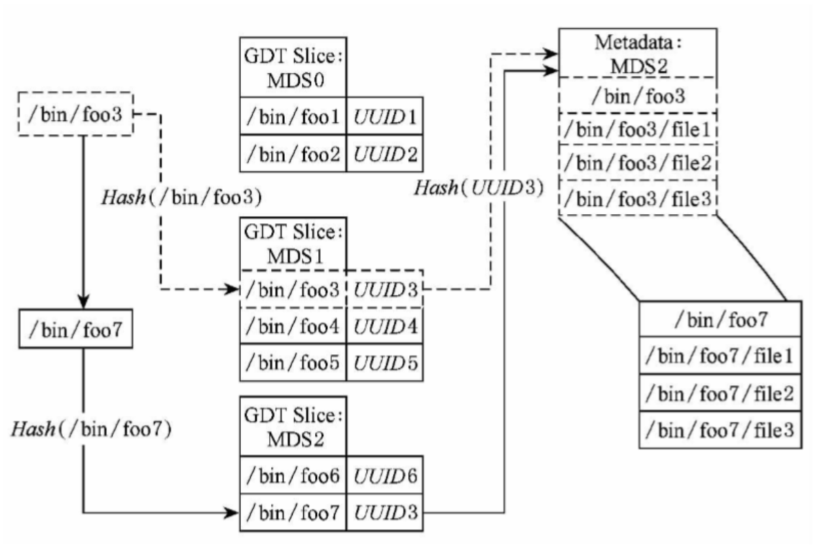

Fig. 2. Operation using GDT

## *3.3. Shared storage pool*

The shared storage pool of Clover mainly solves the problem of consistency in distributed metadata analysis processing. Because of the mapping structure of the Clover two levels, Clover metadata is not stored on the corresponding metadata server, which is possibly from another metadata server. Storing data on a metadata server while operating it on another metadata server are likely to cause metadata differences. Therefore, in order to ensure consistency of metadata in the file system, Clover reaches two protocols with the shared storage pool in response to the two level mapping structure of Clover, so as to ensure that the system can recover quickly when the metadata failed or varied.

There are the preparatory stage and the execution phase of the two paragraphs of the agreement. In the preparation phase, the client proposes requirements, and the system selects the metadata server according to the mapping structure of the two levels and sends the ready commands to them. When the selected server receives the ready command, the censoring itself is carried out to ensure that it meets the requirements and can complete the commands. At this point, in the shared storage pool, the status of these metadata servers is generated, namely, the original data files.

When the selected metadata server reviews itself, the system is ready to complete the instructions, while the metadata server that does not pass the audit will not send ready to complete commands. When the selected metadata server sends the prepared instruction and the system receives it, the preparation phase is complete. When the system does not receive the ready and complete command of the selected metadata server within a specified time, the preparation is not completed. At this point, in the shared storage pool, records are made to metadata servers that are not ready for replies, at the same time, the system sends the failed records to the metadata server.

When the preparation phase is complete, the system enters the execution phase. As the execution phase begins, the system assigns roles to the underlying metadata server and issues different execution commands. At the same time, the shared storage pool will back up the relevant metadata server records. After the metadata server executes the relevant commands, the result is returned to the client and the execution of the command is resumed to the system. When the execution fails, the metadata server sends the canceled command to the system, and in the shared storage pool, a failed record is added to the metadata server's record, and the system sends a failed record to the metadata server.

### *3.4. Metadata recovery mechanism*

In the face of huge amounts of data, it is inevitable that the data server will fail. With the expansion of Clove cluster size, the number of metadata increases with the geometric index, and metadata is used more frequently in the process of metadata processing and analysis. However, with the frequent use of metadata, there are failures, errors and other phenomena. How to recover metadata, optimize the system, and avoid the metadata server failures and node failures because of the storage errors or invalid metadata has become the focus of the research.

The Clover cluster is in the shared storage pool, and the metadata server is used for each time, backups and records are automatically generated. When the metadata server fails or fails to respond, backups from the shared storage pool play a role and operate on the associated records. In this way, both the system can be optimized and the metadata for invalidation or error can be recorded. In Fig. 3, when the 1 command execution system metadata server fails, the backup stored in a shared storage pool plays a role in the system and avoids no response.

## 4. Result analysis and discussion

### *4.1. Operational performance of metadata*

When metadata was operated under a Clover cluster, its performance was tested. These operations such as file information capture, creation, deletion, renaming, and subdirectory creation, may occur on a metadata server, or may be linked in series with several metadata servers and completed through multiple nodes. When the test was carried on, loads were added to the Clover cluster, files were created on

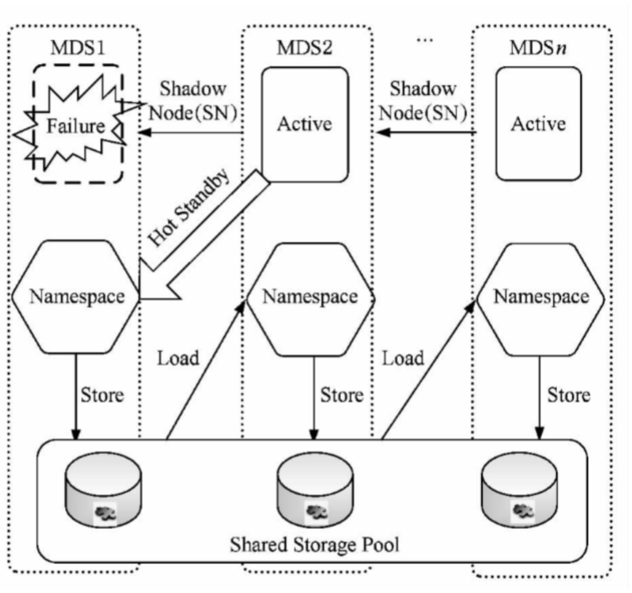

Fig. 3. Backup based on shared storage pool

each metadata server, and then the properties of the file were searched, and a series of operations such as creating sub files, renaming, deleting and so on were carried out, so as to enable the system to begin processing large amounts of data. For each of these operations, each operation was performed one hundred thousand times. The performance test results are shown in Fig. 4. Vertical coordinates showed the time taken by operations, which was used to measure the performance of metadata. Clover-nMDS stood for the number of metadata servers in the Clover cluster.

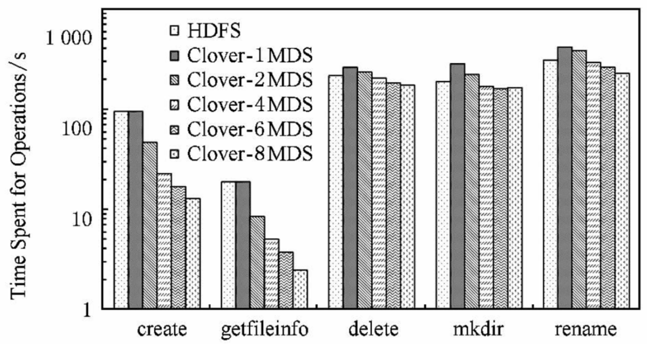

Fig. 4. Results chart of metadata operational performance testing

As can be seen from Fig. 4, under the condition of a metadata server for HDFS and Clover, there was no difference between the time required for file creation and information acquisition. However, with the increasing number of metadata servers, the time required for file creation and information acquisition became shorter and shorter. Because with the increasing number of metadata servers in the Clover cluster, metadata transfers did not occur between each other, so that the time required to complete the operation became shorter and shorter. For file deletion, renaming, and the creation of sub files, the 1–2 metadata servers did not have any advantages over HDFS, but consumed more time instead. However, as the number of metadata servers increased, the time required for operations began to decrease.

## *4.2. Metadata extension performance testing*

The extended performance of metadata is a key performance of large data processing and analysis. In the case of using a constant number of clients to operate together, HDFS was taken as a reference index, the metadata extension performances of Clover clusters with different metadata servers were compared in this paper. The number of clients should be more than 10, and twelve clients were selected for simultaneous operation.

Table 1. Metadata extension performance of different Clover metadata servers

| Number of MDS Added | Create | Getfileinfo | Delete | Mkdir | Rename |
|---|---|---|---|---|---|
| 1 | 159.32 | 156.45 | –3.92 | –9.35 | –7.73 |
| 3 | 112.38 | 147.44 | 5.13 | 6.33 | 5.06 |
| 5 | 103.56 | 136.92 | 6.51 | 8.41 | 8.39 |
| 7 | 100.73 | 116.59 | 7.7 | 9.63 | 9.17 |

As can be seen from Table 1, as a large number of 10 clients operated simultaneously, the number of metadata servers in the Clover cluster increased, the performance of the element boards became better and better. For file creation and file information access, adding a metadata server had maximum scalability, which was the same as the previous metadata performance test results. However, for the metadata extension of file deletion, renaming and the creation of sub files, the more the number of metadata servers increased, the better the scalability was achieved. Because in the presence of multiple metadata servers, load metadata servers became the same, and were no longer focused on one or several metadata servers, the extensibility of metadata was improved.

## *4.3. Metadata usability testing*

The availability of metadata refers to the time which is taken to recover a system when the metadata server fails. HDFS itself has a large number of spare nodes, and can detect the metadata server through periodic disk data detection, heartbeat detection and so on. When the metadata server fails, it is necessary to manually load the relevant files, check the metadata server and node, and restore the system through the standby nodes. The Clover cluster supports the backup and recovery of the metadata server through the backup data of synchronous metadata server in the shared storage pool and through the system failure mechanism. Table 2 shows the system recovery time required by the HDFS and Clover clusters for different size dependent recovery files under the condition of a metadata server.

Table 2. Size and recovery time of the file

| Size of the Checkpoint File/MB | HDFS | Clover |
|---|---|---|
| 16 | 3.334 | 0.728 |
| 32 | 6.472 | 0.871 |
| 64 | 12.975 | 0.649 |
| 128 | 25.832 | 0.732 |
| 256 | 52.187 | 0.806 |

As can be seen from the table, the recovery time required by the Clover cluster was significantly smaller than that of HDFS, and the size of the recovery file was not related to the size of the Clover cluster as compared with the HDFS. The reason is that in the Clover cluster, the shared storage system can also backup metadata servers, so that no matter how large the file size is restored, the Clover cluster can always recover the system quickly and keep the service of the file system uninterrupted.

## 5. Conclusion

With the advent of the big data era, the processing and analysis of mass data become more and more common. In this paper, the key technologies of distributed system design for big data were studied, and the Clover cluster system based on the two-level mapping structure of the global distributed directory Hash table was proposed. With the backup and record of the shared storage pool as the metadata server, the system was optimized by using the two section decision method, and a stable and efficient file distributed system was obtained. Through related performance testing, it can be found that the Clover cluster system has better metadata operation, metadata expansion is also improved, the system stability is high, and the system optimization is better, thus providing a critical technology for distributed file systems for big data analysis.

With the above advantages, Clover cluster can ensure the operability of metadata, improve its scalability and availability, and lead to more stable and effective distributed file system. With the future optimization and improvement of Clover cluster, Clover cluster technology will occupy a more important position in the key technology centers of big data oriented distributed files.

## References

[1] Y. Wu, F. Ye, K. Chen, W. Zheng: *Modeling of distributed file systems for practical performance analysis.* IEEE Transactions on Parallel and Distributed Systems *25* (2014), No. 1, 156–166.

[2] J. S. Kim, K. Y. Whang, H. Y. Kwon, I. Y. Song: *PARADISE: Big data analytics*

*using the DBMS tightly integrated with the distributed file system.* World Wide Web *19* (2016), No. 3, 299–322.

[3] P. Mali, O. Kulkarni, V. Shah, K. Paste, M. Bhanushali: *Public sharing file system using google drive.* International Journal on Recent and Innovation Trends in Computing and Communication *3* (2015), No. 3, 1550–1553.

[4] R. Malhotra, J. M. Mellichamp: *Intelligent simulation code generator: A relational database management approach.* Journal of Intelligent Manufacturing *8* (1997), No. 2, 125–136.

[5] J. DeBrabant, A. Pavlo, S. Tu, M. Stonebraker, S. Zdonik: *Anti-caching: A new approach to database management system architecture.* Journal Proceedings of the VLDB Endowment *6* (2013), No. 14, 1942–1953.

[6] S. A. Gebrezgabher, M. P. M. Meuwissen, A. G. J. M. O. Lansink: *A multiple criteria decision making approach to manure management systems in the Netherlands.* European Journal of Operational Research *232* (2014), No. 3, 643–653.

[7] X. Hua, H. Wu, S. Ren: *Enhancing throughput of Hadoop distributed file system for interaction-intensive tasks.* Journal of Parallel and Distributed Computing *74* (2014), No. 8, 2770–2779.

[8] N. Cao, Z. H. Wu, H. Z. Liu, Q. X. Zhang: *Improving downloading performance in Hadoop distributed file system.* Journal of Computer Applications *30* (2010), No. 08, 2060–2065.

[9] Y. S. Jeong, Y. T. Kim: *A token-based authentication security scheme for Hadoop distributed file system using elliptic curve cryptography.* Journal of Computer Virology and Hacking Techniques *11* (2015), No. 3, 137–142.

[10] W. W. Smari, S. Fiore, C. Trinitis: *Recent developments in high-performance computing and simulation: distributed systems, architectures, algorithms, and applications.* Concurrency and Computation, Practice and Experience *27*, (2015), No. 9, 2191–2195.

[11] J. Zhou, W. Wang, D. Meng, C. Ma, X. Gu, J. Jiang: *Key technology in distributed file system towards big data analysis.* Journal of Computer Research and Development *51* (2014), No. 02, 382–394.

[12] H. C. Hsiao, H. Y. Chung, H. Shen, Y. C. Chao: *Load rebalancing for distributed file systems in clouds.* IEEE Transactions on Parallel & Distributed Systems *24* (2013), No. 5, 951–962.

[13] A. Mei, L. V. Mancini, S. Jajodia: *Secure dynamic fragment and replica allocation in large-scale distributed file systems.* IEEE Transactions on Parallel & Distributed Systems *14* (2003), No. 9, 885–896.

[14] S. Miltchev, J. M. Smith, V. Prevelakis, A. D. Keromytis, S. Ioannidis: *Decentralized access control in distributed file systems.* ACM Computing Surweys 40 (2008), No. 3, 1–30.

[15] C. K. Lee, H. A. Eagles, J. R. Caradus, K. F. M. Reed: *Investigation of yield and persistence of white clover using cluster analyses.* Euphytica *72* (1993), No. 3, 219–224.

Received July 12, 2017