ATLANTIS PRESS

Advances in Intelligent Systems Research (AISR), volume 142

2017 International Conference on Information Technology and Intelligent Manufacturing (ITIM 2017)

Large-Scale Data Storage and Management Scheme Based on Distributed Database Systems

Qiao SUN, Bu-qiao Deng, Lan-mei FU
Beijing GuoDianTong Network Technology Co., Ltd.,
Beijing, China
e-mail: littlesmart@yeah.net

Zhi-qiang Wang, Xu-bin Pei
State Grid Zhejiang Electric Power Company Information &
Telecommunication Branch,
Zhejiang, China
Jia-song SUN
E. E. Department, Tsinghua University,
Beijing, China
e-mail: sunjiasong@tsinghua.edu.cn

***Abstract*—the existing distributed database management system can realize the data storage with high access bandwidth through the cluster, which has the characteristics of reliable data replication and fault detection and fast automatic system recovery. However, with the existing network platform software and computing model, there is still a need to establish a virtual organization that can implement specific requirements based on the data manipulation mechanism and data access patterns in the network. This paper proposes a large-scale data storage and management scheme based on distributed database, which is more suitable for the use and control of large-scale data access and other types of application and data access modes as needed, compared with other technical solutions. Mode can cover multiple orders of magnitude of data exchange and data processing and input / output systems. The experimental results show that this scheme has the advantages of good reliability, easy operation and support of large amount of data processing.**

Keywords—Distributed database systems; Large-scale data storage; Heterogeneous equipments; Large-scale data processing;

I. INTRODUCTION

Distributed database file management system for clustering through high access bandwidth under the data storage, with reliable data replication and fault detection and fast automatic system recovery and so on. The low latency and high throughput of the distributed management system is well suited for batch processing, while implementing a simple data consistency model to achieve a high-throughput, high-throughput data (typically multiple of 64MB) take/read and write. The portability of distributed database file management systems enables heterogeneous hardware devices and heterogeneous software platforms to be better integrated [1, 2, 3].

The distributed database file management system provides the data location to identify the operating environment and is responsible for working on the nodes where the data resides [4]. The distributed database file management system needs to use a programming model for handling large data sets that are suitable for handling distributed content, especially in standard clusters, through independent nodes. The nodes are usually responsible for managing the mapping between data blocks and their location in the distributed system, by copying the data blocks to multiple machines to implement the fault tolerance mechanism; the other node is usually responsible for assigning tasks and monitoring the execution flow. Distributed database file management system not only need to deal with the input data partition, but also need to arrange cross-node tasks, and management of communication between nodes. In addition, the node needs to be processed at runtime to terminate the task by restarting the node, and the backup data is called when the data is not available.

The establishment of storage systems based on distributed database file management systems is a natural choice for many CPU-intensive and data-intensive networks, based on a number of structural features suitable for distributed computing: high reliability, scalability, portability, throughput the ability to build a platform, and the low maintenance costs. But the integration with existing network platform software and computing models is still difficult [5, 6]. It is necessary to establish a virtual organization capable of realizing specific requirements according to the data manipulation mechanism and the data access mode in the network.

A typical network database file management system has the following characteristics [7,8]: there is a reliable support model to meet the expectations of computing infrastructure reliability, availability and security; can interface with a variety of existing data transmission systems Compatible with the transmission technology; in any hardware component failure recovery process, the device must have a good definition and reliable behavior and behavior is tested and recorded; there is a clear and reliable way to copy the data to prevent it from being in any A single hardware on the loss; there is a reliable way to remove the hardware from the cluster and to ensure that there is no data loss in the process; must have a good interface to monitor the system, if any hardware or software failure to notify.

This paper proposes a large-scale data storage and processing scheme based on distributed heterogeneous device management. It is more suitable for using and controlling large-scale data access than other technologies. It can deal with various types of applications and data as needed Access mode, which can cover multiple orders of magnitude of data exchange and data processing and input/output systems.

The structure of this paper is as follows: firstly, the basic architecture and principle of the distributed management system are introduced; then, the architecture and design of the data storage and processing scheme based on distributed heterogeneous devices are presented; the test results of the scheme are given on the large-scale data set; finally, we give the summary and prospect.

II. Distributed Database File Management System

In the streaming data access mode to store large database files, to read the entire data set of experimental delay is important than to read the first record of the time delay which is not required to executive on the expensive and highly reliable hardware equipment. Distributed database file management system consists of the following structure [9, 10]:

A distributed database file management system consists of a single node and multiple data nodes; the management system has an explicit file system namespace and allows users to store data in a file; inside management system, a file is divided into one or more data Blocks, which are stored in a set of data nodes.

Name nodes is responsible for managing the namespace of the file system, regulating the way customers access files, and executing file system namespace operations. This also determines the mapping of the data block to the data node. By keeping all the files and folders of the metadata in a file system tree. This information is also saved on the hard disk as the following files: namespace image and edit log. Also save a file, including data blocks and data nodes distributed. This information is not stored on the hard disk, but when the system starts from the data node collection.

Each cluster usually have data nodes for managing the nodes used to store. These nodes also handles requests for clients that read and write file systems and perform tasks such as creating, deleting, and copying from nodes. The client or nodes can request to write or read data blocks to the data nodes. Nodes periodically report its stored data block information to nodes. And periodically merge the nodes namespace image file and the modification log to prevent the log file from being too large.

In order to create a virtual organization that can provide I / O and other data services, the following functions need to be integrated with management: Linux interface allows the file system to be written to the user control and provides POSIX interface to management; FTP interface can be integrated as an asynchronous transmission package and write management which is in multi-stream transfer mode to achieve file transfer performance optimization of one of the ways; other interfaces, such as SRMv2, can be accessed into management system.

At the same time, according to the specific needs of the network computing strategy, management provides optional cascading components.

III. Data Storage Scheme with Distributed Heterogeneous Equipment

Based on the above architecture, in order to be able to undertake the access from large-scale data access and heterogeneous systems and heterogeneous hardware equipment and provide multiple orders of magnitude of data exchange and data processing and input/output system, this paper proposed a new distributed database file management architecture for the large-scale data storage and processing program, which was shown in Figure 1. This architecture supports the FTP interface and support other interfaces; in the metadata node support Linux interface, support FTP interface; in the data node only need to support the Linux interface.

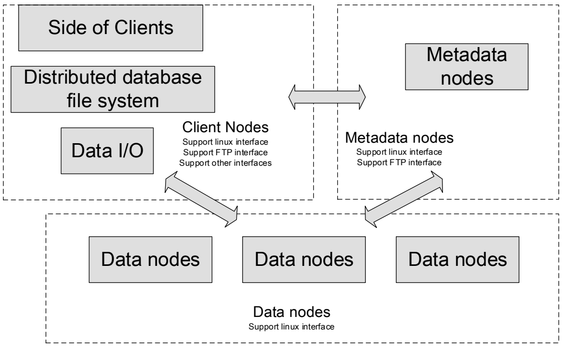

Fig. 1. Large-scale data storage architecture based on distributed heterogeneous equipment.

The basic structure of this approach is:

- ✓ Data storage:

Data storage can be done directly from the client node, metadata node and data node via the linux interface. The support of this method depends largely on the configuration of the URL because it is not certified and authorized by the GSI. The data can also be sent directly to the client node, the method is similar to sending through the Linux interface, but only the file transfer.

- ✓ Data read:

FTP interface can be used in the client node and metadata node for data read, the general use of standard data read mechanism for the transfer of data between two FTP; can also use the Linux interface for the application of simple Data and file access, but this data access is limited to a single Linux interface, it is more suitable for interactive access to data in the local user applications; data can also be sent directly to the client node, but there are some data security problem.

- ✓ System Security:

System security is required when file creation/deletion and creation/deletion of files from a public host. These operations can be done through the local Linux interface, because these operations belong to the local domain management. To ensure security, if you do not use HDFS local user or group account, through the Linux interface or Hadoop remote direct operation of the file is prohibited. Will further provide user authentication and data encryption to improve security in the future.

✓ Cluster Computing: The performance of cluster computing depends primarily on the scalability of the data storage scheme and the heterogeneous hardware device to calculate the load balance between the models; the solution for the same cluster is designed for distributed and shared models, since this mode is most Flexible and flexible, can serve multiple groups of users, and adapt to a variety of applications, data access mode.

IV. EXPERIMENTS AND TEST RESULTS

In order to evaluate the performance of the above method, we evaluated and tested in the existing intranet environment. The configuration of the experiment is:: twenty-five servers IBM-x3650M4-2U (2CPU, 6-core 12-thread Xeon E5-2620, 48G memory) and separated being as metadata/data node, the operating system is Ubuntu11.10 with pre-installed FTP services.

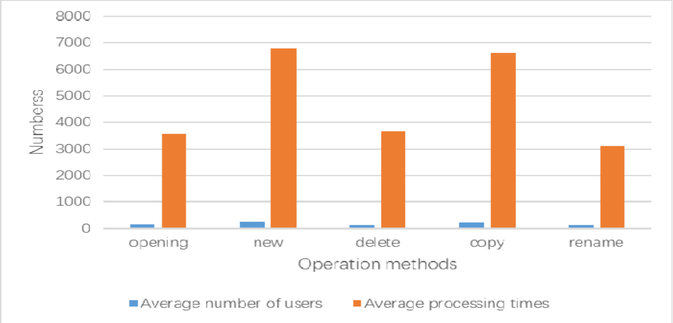

Fig. 2. Data nodes operation statistics.

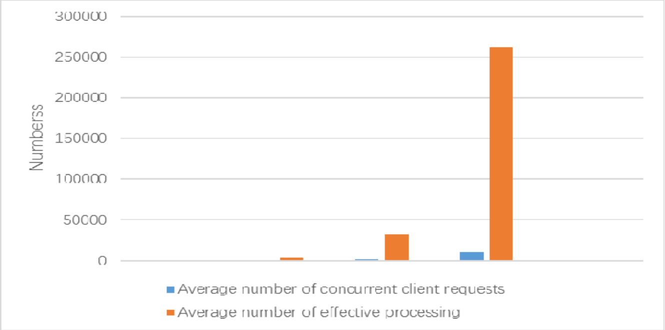

Fig. 3. Average number of concurrent client requests and average effective processing rate.

At one same time, visiting users were about thirty local users. We counted the number of data nodes in the network environment within 96 hours. The results are shown in Fig.2. The number of concurrent client requests is compared with the effective processing rate. The results are shown in Fig.3.

Finally, the round-trip delay generated by the interface type is tested to measure the delay which required to send a message to the response information at different interfaces. The results are shown in Table 1. It shows the speed of processing information between different interfaces is closed.

TABLE I. INTERFACE TYPE AND ROUND TRIP DELAY.

Interface type	Round trip delay（msec）
Linux Interface vs Win Interface	14
Linux Interface vs FTP Interface	12
Linux Interface vs other Interface	17
FTP Interface vs Win Interface	15
FTP Interface vs other Interface	12
Win Interface vs other Interface	13

V. CONCLUSIONS

This paper constructs a set of data storage and processing scheme based on distributed heterogeneous devices. The experimental results show that the proposed large-scale data storage and processing scheme based on distributed heterogeneous devices has the advantages of good reliability, easy operation and support for large-scale data processing. The performance of the data is based on the distributed heterogeneous device Most of the deployment commands are simpler, fewer commands are made by the administrator, fewer system and database file recovery due to hardware or software failures, and support for a large number of simultaneous read and write operations, etc.

The next step will focus on further improving the efficiency of system security, focusing on how to further reduce deployment and maintenance costs.

ACKNOWLEDGEMENT

This research was financially supported by Science and Technology Project of the State Grid Corporation of China (SGZJ0000BGJS1500433) and the State Grid Information & Telecommunication Group CO., LTD. (SGITG-KJ-JSKF [2015]0003).

REFERENCES

[1] Howard J H, Kazar M L, Menees S G, et al. Scale and performance in a distributed file system [J]. ACM Transactions on Computer Systems (TOCS), 1988, 6(1): 51-81.

[2] Nishikado T, Kondo M, Fukuoka K, et al. File management system for permitting user access to files in a distributed file system based on linkage relation information: U.S. Patent 5,012,405[P]. 1991-4-30.

[3] Weil S A, Brandt S A, Miller E L, et al. Ceph: a scalable, high-performance distributed file system[C]. OSDI'06 Proceedings of the 7th symposium on operating systems design and implementation, Berkeley, CA. 2007.

[4] Hashem I A T, Yaqoob I, Anuar N B, et al. The rise of "big data" on cloud computing: Review and open research issues [J]. Information Systems, 2015, 47: 98-115.

[5] Coronel C, Morris S. Database systems: design, implementation, & management [M]. Cengage Learning, 2016.

[6] Fan W, Bifet A. Mining big data: current status, and forecast to the future [J]. ACM sIGKDD Explorations Newsletter, 2013, 14(2): 1-5.

[7] Boyle W B, Fallone R M. System and method for optimizing garbage collection in data storage: U.S. Patent 8,706,985[P]. 2014-4-22.

[8] Cordray C, Link D, Chart R, et al. Self-configuring network management system: U.S. Patent 9,077,611[P]. 2015-7-7.

[9] Howard J H, Kazar M L, Menees S G, et al. Scale and performance in a distributed file system [J]. ACM Transactions on Computer Systems (TOCS), 1988, 6(1): 51-81.

[10] O. O'Malley, K. Zhang, S. Radia, R. Marti, and C. Harrell, Hadoop security design, Yahoo, Inc., Tech. Rep, 2009.

[11] Kuo M H, Sahama T, Kushniruk A W, et al. Health big data analytics: current perspectives, challenges and potential solutions[J]. International Journal of Big Data Intelligence, 2014, 1(1-2): 114-126.

[12] Chang F, Dean J, Ghemawat S, et al. Bigtable: A distributed storage system for structured data [J]. ACM Transactions on Computer Systems (TOCS), 2008, 26(2): 4.

[13] Thekkath C A, Mann T P, Lee E K. Scalable distributed file system: U.S. Patent 6,173,293[P]. 2001-1-9.

[14] Yalagandula P, Dahlin M. A scalable distributed information management system [M]. ACM, 2004.

[15] Baker M G, Hartman J H, Kupfer M D, et al. Measurements of a distributed file system[C]. ACM SIGOPS Operating Systems Review. ACM, 1991, 25(5): 198-212.

[16] Ma Y, Wu H, Wang L, et al. Remote sensing big data computing: Challenges and opportunities[J]. Future Generation Computer Systems, 2015, 51: 47-60.