International Journal of Computer Networks and Communications Security

VOL. 3, NO. 5, MAY 2015, 229–234
Available online at: www.ijcncs.org
E-ISSN 2308-9830 (Online) / ISSN 2410-0595 (Print)

A REVIEW: Distributed File System

Shiva Asadianfam[1], Mahboubeh Shamsi[2] and shahrad kashany[3]

[1, 3] Department of Computer Engineering, Qom Branch, Islamic Azad University, Qom, Iran

[2] Department of Electrical & Computer Engineering, Qom University of Technology, Qom, Iran

E-mail: [1]sh_asadianfam@yahoo.com, [2]mahboubehshamsi@yahoo.com, [3]shahrad.kashany@ustmb.ac.ir

ABSTRACT

Data collection in the world are growing and expanding. This capability, it is important that the infrastructure must be able to store a huge collection of data on their number grows every day. The conventional methods are used in data centers for capacity building, costs of software, hardware and management of this process will be very high, today. File system architecture is that, independent of the data stored in its metadata. All of Distributed File System (DFS) are used for processing, storing and analyzing very large amount of unstructured data. The most common file system can be noted Google File System (GFS) and Hadoop Distributed File System (HDFS). In this paper, we present a review on design of distributed file system. File systems compared by scalability, availability, compatibility, extensibility, autonomy and etc.

Keywords: *Distributed File System, Big Data, Design of DFS, Hadoop Distributed File System(HDFS), Google File System(GFS), Review of DFS.*

1 INTRODUCTION

In these years, increasing number of organizations are facing the problem of explosion of data and the size of the databases used in today's enterprises has been growing at exponential rates[1]. Data is generated through many sources like business processes, transactions, social networking sites, web servers, etc. and remains in structured as well as unstructured form [2].

The need for such a level of scalability and increasing volumes of data center storage environment and the need for sustainable, large companies Web-based service provider has to meet your needs, choose another type of storage media systems and distributed file management based on the storage media of object-based. Distributed File System (DFS) is a set of client and server services that allow an organization using Microsoft Windows servers to organize many distributed SMB file shares into a distributed file system. DFS provides location transparency and redundancy to improve data availability in the face of failure or heavy load by allowing shares in multiple different locations to be logically grouped under one folder, or DFS root [3]. Such as, Chien-Ming Wang, Chi-Chang Huang and Huan-Ming Liang [7] are presented a distributed file system, called ASDF. That is an autonomous and scalable distributed file system. Lubos Matejka and et al., [9] are presented a distributed file system, which is named KIV-DFS. That is suitable for mobile devices. In this article, we present a review on design of distributed file system.

This paper structured is follows. In Section 2, the background information required for the better understanding of DFSs presented in this paper is discussed. In this section, first the precise definition of the Distributed File System (DFS) is expressed. The most common Distributed File System architecture are explained, and in Section 3, the studies in design of DFS are presented. Finally, in Section 4, the general conclusion are discussed.

2 BACKGROUND

In this section, the background of distributed file system, popular DFS and a brief description of each DFS are presented.

2.1 Design Goals of DFS

Data collection in the world are growing and expanding. It is important that the infrastructure must be able to store a huge collection of data on their number grows every day. To integrate massive distributed storage resources for providing large storage space is an important and challenging issue in Cloud and Grid computing [7].

Distributed file systems may aim for "transparency" in a number of aspects. That is, they aim to be "invisible" to client programs, which "see" a system which is similar to a local file system. Behind the scenes, the distributed file system handles locating files, transporting data, and potentially providing other features listed below [4]:

- Access transparency is that clients are unaware that files are distributed and can access them in the same way as local files are accessed.
- Location transparency; a consistent name space exists encompassing local as well as remote files. The name of a file does not give its location.
- Concurrency transparency; all clients have the same view of the state of the file system. This means that if one process is modifying a file, any other processes on the same system or remote systems that are accessing the files will see the modifications in a coherent manner.
- Failure transparency; the client and client programs should operate correctly after a server failure.
- Heterogeneity; file service should be provided across different hardware and operating system platforms.
- Scalability; the file system should work well in small environments (1 machine, a dozen machines) and also scale gracefully to huge ones (hundreds through tens of thousands of systems).
- Replication transparency; to support scalability, we may wish to replicate files across multiple servers. Clients should be unaware of this.
- Migration transparency; files should be able to move around without the client's knowledge.

2.2 The Most Common DFS

Google, one of the Internet giants that scalability problems and issues facing storage media and the creation of a distributed file system, the Google engineers presented a solution to the problem in 2003 [5]. Google File System (GFS) describe documents that have resulted in a much broader distributed file system applications is highly similar to the GFS [6]. The system file named Apache Hadoop Distributed File System (HDFS) has been introduced [6].

2.2.1 Google File System [5]

Google tends to store large data files and application files using "producer-consumer queues", is used. Google, more details of the technical architecture of GFS is kept confidential for obvious reasons. But in an article published by Sanjay Ghemawat, a member of the Google team, Howard Gobioff, based engineer and Shun-Tak Leung was a member of the senior engineers in 2003, Google File System (GFS) has been specially designed keeping in mind the priorities. The aim of this paper is that the GFS, conversion of a large number of servers and hard disk drives for low-cost, to a range of hundreds of terabytes of data that can be stored and managed, and in the event of a fault or hardware failures can solve the problem there. The next element of its architecture servers provide a "chunk servers" are known.

2.2.2 Hadoop Distributed File System [6]

The Hadoop Distributed File System (HDFS) is a distributed file system providing fault tolerance and designed to run on commodity hardware. HDFS provides high throughput access to application data and is suitable for applications that have large data sets. Hadoop provides a distributed file system (HDFS) that can store data across thousands of servers, and a means of running work (Map/Reduce jobs) across those machines, running the work near the data. HDFS has master/slave architecture. Large data is automatically split into chunks which are managed by different nodes in the Hadoop cluster.

3 STUDIES IN DISTRIBIUTED FILE SYSTEM

Much research in various designs has been performed. In this article, just several cases are

presented. In this section, the focus is on those tasks undertaken for unstructured data, small image storing, and advantages of these designs of distributed file system. So, First P2P, Client/Server technologies explain in DFS design and then DFS designs based on HDFS will presented. Table 1 summarizes the kind of designs of distributed file system and advantages related to present work.

3.1 Related studies on P2P technology for DFS design

Chien-Ming Wang, Chi-Chang Huang and Huan-Ming Liang [7] are presented a distributed file system, called ASDF. The proposed system consists of three parts: the core, client and services. While sharing many of the same goals as previous distributed file systems such as scalability, availability, reliability, and performance, their system is also designed with an emphasis on compatibility, extensibility and autonomy. They said, ASDF will be useful in Cloud and Grid computing.

3.2 Related studies on Client/Server technology for DFS design

The proposed file system by Tzong-Jye Liu, Chun-Yan Chung, Chia-Lin Lee [8] stores the file replications in different network area. They also design a load detection module in the proposed system. The module will detect the load of each file servers and improve the overall system performance [8]. Lubos Matejka and et al., [9] are presented a distributed file system, which is named KIV-DFS. Google file system are not suitable for mobile devices. The system is based on the client-server architecture. The client module allows the users to work with data. In their design and implementation, this module appears in three versions:

- ✓ As a standalone program.
- ✓ As a core module of the operating system.
- ✓ As a FUSE module.

The client communicates with the server module. The client can be connected by any kind of network providing a TCP/IP stack [9]. KIV-DFS increase the throughput of the system, the security of the data and improves the service availability in large heavily loaded [9].

In the paper [10], is presented a distributed file system, called QFS. That consists of two major components: the management node (master), storage node (chunk). Management node is the central node, its main role is load balancing and scheduling. The management nodes record the information such as status information of grouping and chunk server, and it doesn't include the information of file index, so the amount of memory that is occupied is relatively less. In addition, when the client (application) and the chunk server access the master server, the master server scan the information of grouping and chunk server in the memory, and then gives the answer [10].

In the paper [11], are presented a distributed file system. In this system, Name Server is responsible for data classification and allocation of storage. Name Server used to support the Name Server works. Data Server placed under the Name Server, in a Data server is a type of image to save it. Data Server are not related, and do not know what kind of video is stored on others. To manage images, a fundamental approach to describe the data and then use the descriptive information for the implementation of the operation. Raw data, basic features, low-level features and semantic features are used to describe the image data. A tree structure is used to display image data. The root node, raw data refers to data files stored images. In the middle child of the root node, low-level features and key characteristics of states. Low-level features include image data features, such as color, texture, shape of images. Basic features include attributes such as name, type, author, and creation time. The leaf node, using the expression of semantic features that include good writer, explaining the topic, and a low-level features.

3.3 Related studies on DFS design Based on HDFS

Konstantin Shvachko and et al., [12] are peresented a distributed file system, called HDFS. HDFS architecture is very close to the GFS. It will follow three layer structure and single reference. Each cluster or clusters of Apache Hadoop, a reference server is called NameNode. This server is responsible for tracking metadata and address of copies of each of the blocks that hold 64 MB of data. NameNode copy the data in cluster and systems are responsible for writing and reading data on the Data Node responsible. By default, each data block three times Copy (Data Replication) is:

- ✓ First block is written in current server.
- ✓ In a current server on a same cluster.
- ✓ In a server on another cluster.

But, you can change the settings of the clusters, then the number of copies to be changed. This has two major advantages:

First you against hardware failures, such as the burning of the hard disk, server hardware problems and are safe. If any of the servers for reasons outside the network, the data on other servers are called. The second advantage of this feature is that you no longer need to use the RAID technology (strategies for data replication), and you can use up your hard drive space.

In the paper [13], present a NameNode cluster file system based on HDFS, called Clover. This file system exploits two critical features: an improved 2PC protocol which ensures consistent metadata update on multiple metadata servers and a shared storage pool which provides robust persistent metadata storage and supports the operation of distributed transactions [13]. Further experimental results show their system can achieve better metadata expandability ranging from 10% to 90% by quantized metrics when each extra server is added, while preserving similar I/O performance [13].

hatS [14] is a novel redesign of HDFS [11] into a multi-tiered storage system. An important difference between hatS and HDFS is the design of the DataNode. In HDFS [11], each participating node hosts a single DataNode instance, constituting multiple storage devices, regardless of their characteristics such as supported I/O rates and capacity. In contrast, each participating node in hatS hosts multiple DataNode instances, where each instance represents only one type of storage device.

Liu Jiang and et al., [15] proposed the optimization of file system which provides a better performance based on small files. Farag Azzedin [16] proposed a similar architecture with HDFS, but NameNode is distributed.

Table 1: Overview of the 10 DFS studies

	Article Name	Author (s)	Publish	Year	Pages	Advantages	Name of DFS
P2P	ASDF: An Autonomous and Scalable Distributed File System	Chien-Ming, Wang, Chi-Chang Huang, Huan-Ming Liang	11th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing	2011	485-493	Scalability, Reliability, Compatibility, extensibility , autonomy	ASDF
C/S	A High Performance and Low Cost Distributed File System	Tzong-Jye Liu, Chun-Yan Chung, Chia-Lin Lee	IEEE	2011	47-50	low cost, reliability, scalability, improve the overall system performance	-
	Distributed File System with Online Multi-Master Replicas	Luboš Matějka, Jiří Šafařík, Ladislav Pešička	Second Eastern European Regional Conference on the Engineering of Computer Based Systems	2011	13-18	Scalability proper for mobile devices, Increase throughput, Increase the security of the data, improves the service availability	KIV-DFS

C/S	A Kind Of Distributed File System Based On Massive Small Files Storage	DI LIU, SHT-JIE KUANG1	IEEE	2012	394-397	improve metadata management, high concurrency, storage efficiency, simple design, efficient design, storage capacity, scalability	QFS
	Distributed file system and classification for small images	Shaojian Zhuo, Xing Wu, Wu Zhang and Wanchun Dou	IEEE International Conference on Green Computing and Communications	2013	2231-2234	Reduce meta data, High Speed access	-
	The Hadoop Distributed File System	Konstantin Shvachko, Hairong Kuang, Sanjay Radia, Robert Chansler	IEEE	2010	1-10	Scalability, improves the overall availability, all of Goals in DFS design	HDFS
Based on HDFS	Clover: A distributed file system of expandable metadata service derived from HDFS	Youwei Wang, Jiang Zhou, Can Ma, Weiping Wang, Dan Meng, Jason Kei	IEEE International Conference on Cluster Computing	2012	126-134	Expandablity, bottleneck-less, no single point of failure	Clover
	hat*S*: A Heterogeneity-Aware Tiered Storage for Hadoop	Krish K.R., Ali Anwar, Ali R. Butt	14th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing	2014	502-511	Improve throughput , improve job completion time, improve the utilization of the storage devices	hatS
	The optimization of HDFS based on small files	Liu Jiang, Bing Li, Meina Song	the 3rd IEEE International Conference on Broadband network and Multimedia Technology	2010	912-915	Increase the speed of reading, reduce the read request, reduce the write request received by the name node	-
	Towards a scalable HDFS architecture	F. Azzedin	IEEE International Conference on Collaboration Technologies and Systems (CTS)	2013	155-161	Highly available, Widely scalable, fault tolerant,	-

4 RESULTS

Big data is set of data name used to refer much more than that with common software can be obtained in a reasonable time, refine, and manage and process. The concept of "size" of big data is changing continuously and gradually gets larger. So, distributed file system should manage and process this large volume of data. To integrate massive distributed storage resources for providing large storage space is an important and challenging issue in Cloud and Grid computing. In this article, various design of the distributed file system were discussed. A brief description of the use of these Distributed File Systems were provided to achieve our goals by using big data. So, All of DFSs are used for processing, storing and analyzing very large amount of unstructured data.

5 REFERENCES

[1] Aditya B. Patel, Manashvi Birla, Ushma Nair, "Addressing Big Data Problem Using Hadoop and Map Reduce", NIRMA university international conference on engineering, nuicone, 06-08december, 2012.

[2] Impetus white paper, March, 2011, "Planning Hadoop/NoSQL Projects for 2011" by Technologies, Available: http://www.techrepublic.com/whitepapers/planninghadoopnosql-projects-for-2011/2923717, March, 2011.

[3] http://en.wikipedia.org/wiki/Distributed_File_System_%28Microsoft%29

[4] Clustered file system, Available: http://en.wikipedia.org/wiki/Clustered_file_system#Distributed_file_systems, December 2013.

[5] Sanjay Ghemawat, Howard Gobioff, and Shun-Tak Leung, "The Google File System", SOSP'03, Bolton Landing, New York, USA, October 19–22, 2003.

[6] Apache Software Foundation. Official apache Hadoop website, http://hadoop.apache.org/, Aug, 2012.

[7] Chien-Ming Wang, Chi-Chang Huang and Huan-Ming Liang, "ASDF: An Autonomous and Scalable Distributed File System", 1th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing, DOI 10.1109/CCGrid.21, pp. 485-493, 2011.

[8] Tzong-Jye Liu, Chun-Yan Chung, Chia-Lin Lee, "A High Performance and Low Cost Distributed File System", IEEE 978-1-4244-9698-3/11/$26.00 ,pp. 47-50, 2011.

[9] Luboš Matějka, Jiří Šafařík , Ladislav Pešička , "Distributed File System with Online Multi-Master Replicas", Second Eastern European Regional Conference on the Engineering of Computer Based Systems, DOI 10.1109/ECBS-EERC.12, pp. 13-18, 2011.

[10] DI LIU, SHT-JIE KUANG1 , "A Kind Of Distributed File System Based On Massive Small Files Storage", IEEE 978-1-4673-4685-6112/$31.00 , pp. 394-397, 2012.

[11] Shaojian Zhuo, Xing Wu, Wu Zhang and Wanchun Dou, "Distributed file system and classification for small images", IEEE International Conference on Green Computing and Communications and IEEE Internet of Things and IEEE Cyber, Physical and Social Computing, DOI 10.1109/GreenCom-iThings-CPSCom.422, pp. 2231-2234, 2013

[12] Konstantin Shvachko, Hairong Kuang, Sanjay Radia, Robert Chansler, "The Hadoop Distributed File System", IEEE 978-1-4244-7153-9/10/$26.00, pp 1-10, 2010.

[13] Youwei Wang, Jiang Zhou,Can Ma, Weiping Wang, Dan Meng, Jason Kei , "Clover: A distributed file system of expandable metadata service derived from HDFS", IEEE International Conference on Cluster Computing, DOI 10.1109/CLUSTER.54,pp 126-134, 2012.

[14] Krish K.R., Ali Anwar, Ali R. Butt, "hatS: A Heterogeneity-Aware Tiered Storage for Hadoop", 14th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing, DOI 10.1109/CCGrid.51, pp. 502-511, 2014.

[15] Liu Jiang, Bing Li,Meina Son, "The optimization of HDFS based on small files", Proceeding of IC-BNMT2010, the 3rd IEEE International Conference on Broadband network and Multimedia Technology, pp. 912-915, 2010.

[16] F. Azzedin. "Towards a scalable HDFS architecture", in IEEE International Conference on Collaboration Technologies and Systems (CTS), pp. 155-161, 2013.