International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 07 Issue: 05 | May 2020 www.irjet.net p-ISSN: 2395-0072

A Study of Distributed File Systems

Shivam Raj[1], Padukere Tejas Upadhya[2], Suyash Pathak[3]

[1-3]*Department of CSE, Ramaiah Institute of Technology, Bengaluru, India*

Abstract- Distributed system plays an important role in analysis and processing of huge amount of data. It is used for large number of applications such as forecasting, real time data analysis, and various other scientific application. Distributed File Systems has emerged as one of the most effective and convenient solutions to the issue of handling computations with increase in the quantity of data. Distributed file system (DFS) is mainly aimed to enable clients to use a common file system which paves the way for sharing data and resources over physically distributed machines. Many distributed computing projects have tried to implement and design distributed file systems with distinct architectures and have provided with various functionalities with them. This paper tries to build a taxonomy to understand and analyze distributed file system architecture and uses them to describe the implementation of existing distributed file systems.

Keywords - Distributed systems, file systems, distributed computing,

I. Introduction

Distributed File System consist of clients and resources to store data which are spread over a network. It is majorly aimed at providing a feeling of centralized file system in spite of the fact that it has been implemented in a distributed manner. DFS can be considered as one of the examples of distributed systems. It consists of several central servers to store data which is accessed from various remote users of the network. Files in distributed systems are shared among the users in unified and hierarchical fashion. The requirements of distributed file system are totally different if compared with the requirements of the locally stored file system. The speed of data recovery following any form of system failure is one of the major concerns. Thus, the feature provided for fault tolerance should be implemented efficiently. The amount of data handled for any computation is usually preferred to be small in the size of megabytes but the files stored in distributed file system are of sizes more than gigabytes. Some file systems divide the data into blocks to reduce the amount of data handled at once but this call for an additional mapping method which reduces performance. Many files in distributed file system are required to be of write once and read many times type. Therefore, optimized file reader and writer are provided by many file systems. Some of the FS also provide the feature of editing at any arbitrary point in the file. Distributed file systems are required to keep a central node which maintains a record of all the files stored in the system. This is done to avoid the difficulty of traversing large number of nodes to retrieving information for a given file. The paper is divided into five sections. Section II describes several issues related to designing a distributed system. The detailed taxonomy of distributed file system is explained in the third section. The fourth section outlines various existing distributed file systems while section V concludes the paper.

II. Issues in Distributed File System

The various issues related to distributed file systems can categorize in transparency, reliability, security, performance, scalability and flexibility. These issues are described as follows -

A. Transparency

The users are supposed to be unknown of the fact that the system is physically distributed over a network. There are various types of transparencies including access, location, migration, relocation and migration which are described as follows:

Transparency	Description
Access	Hides the differences in data representation and how a resource is accessed.
Location	Hides the physical location of a resource.
Migration	Hides the movement of a resource to another location.
Relocation	Hides the movement of a resource to another location while in use.
Replication	Hides the fact that multiple copies of the resource exist without user's knowledge

B. Flexibility

It is achieved through the use of either monolithic kernel or microkernel on every machine. The kernel mainly deals with the management of memory, resource and process. Monolithic kernels provide all the functionalities without considering the fact if all machines use it or not while micro kernels provide services based on the need of the system.

C. Reliability

It ensures the availability of the data without any discrepancy. Thus, a distributed system is preferred where multiple processes protects from crashes of single processor systems.

D. Performance

It is required for an application to run on a distributed system in the same way it runs on a single system. There are various metrics to measure performance including utilization of system, throughput, response time and uses of network capacity.

E. Scalability

Distributed systems should be designed to handle large number of CPUs. They should also satisfy the need to increase the number of CPUs to include more users and resources.

F. Security

There are three major features of security including confidentiality, integrity and availability. Confidentiality stands for protecting the systems from any kind of unauthorized access Integrity means protecting the data from getting corrupted. Availability ensures throughout accessibility of data in spite of any failure.

G. Fault Tolerance

The user should not be aware of any kind of switching among the processors in case of any server breakdown or system failure.

III. Taxonomy of Distributed File Systems [2]

The analysis of various features related to DFS helps in taking the most suitable and feasible file system. The various factors considered for taxonomy are as follows

A. Architecture

There are various types of DFS architectures including client server architecture, cluster based distributed file system, symmetric architecture, asymmetric architecture and parallel architecture Sun Microsystems's Network System is based on client server architecture to provide a systematic local file system. GFS is a cluster based distributed file system with a master associated with multiple chunk servers. In file systems with symmetric architecture clients provides the metadata manager code which helps nodes to understand the structure of the disk. It basically uses peer to peer based technology. However asymmetric architecture consists of more than one nodes which manages metadata to maintain the file system and the structure of the disk as well. Parallel architecture places blocks of data in parallel across many different storage devices on several storage servers. It provides feature of reading and writing parallely.

B. Processes

The major concern is to keep processes stateful or stateless. The advantage of keeping stateless process is that it is simple. Most of the distributed file provides stateful processes. The stateless architecture provides an advantage of resuming after failure without disturbing the whole system.

C. Communication

The method used for communication in distributed file system Remote Procedure Call (RPC) method is because they make the independent system. RPC approach considers two communication protocols including TCP and UDP. There is one more method to communicate in distributed file system which is based on files. This method uses file like syntax to access all the resources.

D. Naming

The current approach uses a main metadata server to run file name space. The synchronization problem is solved with decoupling of metadata and data which enhances the namespace of the file. Metadata is distributed across all the nodes to understand the layout of the disk.

E. Synchronization

It is required to properly explain the way of reading and writing to avoid any kind of problem on sharing of same file to more than one user. It is also necessary to consider

the file locking system in a distributed file system. Some systems give locks on objects to clients while others perform every operation synchronously on the server.

F. Caching and Replication

Checksum is utilized by large portion of DFS and to approve the information after sending through correspondence organized. An important role is played by caching and replication in DFS as they operate over very large geographical area. There are various ways to achieve that, such as,

1. Client-side caching

Server expects client to query about integrity of the data

2. Serve Side replication

A server tracks all the open file to ensure mutual exclusion. It ensures that no other clients cache the data which is under editing and if no one is writing multiple clients can read the data.

G. Fault Tolerance

It is same as the replication includes in light of the fact that replication is made to give accessibility and real time support mechanism to clients. The two methodologies are failure as an exception and failure as norm. The first one excludes the failed node from the system and perform roll back to the last save point while the latter creates multiple copy of each data is present and such copying is allowed.

IV. OUTLINE OF DISTRIBUTED FILE SYSTEMS [3]

1. NETWORK FILE SYSTEMS (NFS)[4]

In mid of 1980s Silicon valley based Sun Microsystem developed Network File System(NFS). It has become the first choice for distributed file system. NFS has 2 versions: NFSv3 and NFSv4. Implementation of NFS is quite similar to a UNIX system. Hard links and symbolic links is supported for file sharing between users and different directories. Most of the operations on file is supported that is known to be supported by

A. NFS GOALS

The Goals NFS looks to achieve are remote file access and it should be accessed. For remote file access to work in the intended way multi-platform support must be present and in case of a crash due to some external factor there should be crash recovery mechanism built into the server.

B. Architecture

It gives abstract perspective on the nearby system of files. The user accesses the servers using an interface like interface of nearby file system. Caching can be used in client's system to save transfer time and unwanted network traffic. All the operations are defined and performed by the server. Like file system in OS VFS acts as an interface between client and all files on the server.

C. Process

Servers do not retain any information about past requests. Because of being stateless crashes were severe and mitigation became hard. Only mitigation step present was to reboot the system. Client is stateful. In NFS v4 server was made stateful

D. Communication

RPCs are used by the clients to communicate to server. File operation procedure calls are remote. Steps involved in the communication are:

(i) Request received by the server

(ii) parameters are passed to NFS servers and creates the request.

(iii) operation is performed by the VFS layer on local FS.

(iv) Result is delivered to the client.

E. Synchronization

File synchronisation is not necessarily provided if more than two users access the same file. Two methods are provided by NFS v4 for updating the disk: Write through and write back.

F. Replication and Caching

Server Caching works similar to that of caching in UNIX for local files: pages are cached until replacement. For files local to the system, writes are done on 30s interval. In local context server caching is known to work well.

G. Fault Tolerance

It provides fault tolerance in limited scope but it works effectively. Operations are halted in case of server failure. Failure recovery is done through the stateless design.

2. Andrew File system

It came into existence as an offshoot to a larger project named Andrew. AFS was originally developed for computers which was running on OS like UNIX and BDS. AFS provides all functionalities that are provided by NFS such as transparent access to remote shared files. Implemented as two modules that are part of UNIX processes called Vice and Venus.

A. Goals

To provide file system for larger networks.

B. Architecture

Cell is the basic unit for AFS. A cell in AFS is the set of independently administered server and client machines. Cell participates in global andrew file system tree but this is not necessary. Servers and clients is uniquely mapped to a single cell at a particular point of time. A user can make accounts in more than one cells at a given point of time. Home cell is the cell to which client has first logged on to and rest are foreign cells.

C. Processes

Both client and server are stateless

D. Communication

Done over TCP/IP. Communication between two machines is done RPC protocol. The communication protocol is best suited for WANs.

E. Synchronization

Done with the help of server clock. NTP is used over TCP/IP for doing so with UDP as the protocol. One server is chosen and rest of the servers synchronize the clock with the chosen servers.

3. Google File System [5]

In early 2000s google introduced its own file system to cater the growing demand its own data processing requirements. It was built for high data intensive processing jobs. It is built scalable. Fault tolerance is provided. Organised into clusters where each cluster consisted of hundreds of inexpensive storage hardware.

A. Goals

Three primary goals: Performance, reliability, and availability. Additionally it has design related goals that are:

- Storage components should be cheap and massive amount of data can be stored on it.
- System should be capable of processing large number of requests.
- Can store TBs of data.

B. Architecture

Google file system uses architecture involving clusters. The cluster includes a master node who controls all other nodes, multiple proxy servers that provide service to multiple clients. Files are fixed partitioned. 64 bit address is used to uniquely identify a file chunk. 64 bit is provided by the main server when the chunk is created. For ensuring reliability each file copied multiple times on different servers(nodes).

C. Process

The servers are supposed to maintain records related to the requests made and states.

D. Communication

The user needs to send a request to master node. The master node returns requested information to the client who has requested the information. TCP Protocol is used for communication between the master and client nodes.

E. Synchronization

Uses a carefully made function used for locking which can process multiple operations on the same server.

F. Caching and Replication

This replication policy is used to serve two purposes including reliability of data and the presence of data, and improve the proper use of bandwidth in network. It is required to expand chunk replicas across the racks along with the spreading of these replicas across machines.

G. Fault Tolerance

Master node has the capability to redirect all requests for data to the replica of the node if the node goes down until it comes online.

4. Hadoop File System

It is a kind of distributed file system which is based on GFS and is also kind of open source in nature. Hadoop file system is designed to run on inexpensive hardware[6][7]. Hadoop file system has the capacity to store and handle extremely large files. It provides many similar features when compared to other file systems but hadoop FS has the capacity to deliver high throughput and it is highly tolerant to system faults and any kind of other failures.\

A. Goals

• Hadoop file system is supposed to have effective mechanisms to detect and recover from faults because it runs mostly on commodity hardware so hardware failures are frequent

• When computation is being carried out alongside data retrieval hadoop file system is required to reduce traffic in network and increase throughput.

B. Architecture

Hadoop file system is designed on master slave design which consist of a master node called Name node which handles all other slave nodes and is responsible for file system namespace. Apart from name node there are many data nodes managing data storage.

C. Process

Hadoop file system has stateful servers. A record of all the requests made by the clients and the state of the system is maintained.

D. Communication

TCP/IP protocol is used for communications in hadoop file system. A user needs to establish a connection with the master node on the available TCP port.

E. Synchronization

A file in hadoop file system can be written once only but it can be read as many times as needed. A file cannot be updated after creation.

F. Replication and Caching

Hadoop file system has the capacity to store huge files across multiple machines in big clusters. A file is divided into several blocks and then each block is stored on more than one server to achieve fault tolerance. The size of each block is based on the configuration needed for the file. The replication factor is defined when file is created and altered later if needed. The name node or master module takes all the decisions related to blocks and replication of the blocks.

G. Fault Tolerance

The hadoop sytem's placement method places a copy of file on a node in the local space and places other copy on a completely new node within the local space, and the last copy is placed in the node which is in a completely different space. Therefore by using the replica placement policy, fault tolerance is obtained.

5. Ceph File System

Ceph File System[9] is an open source file system developed by Sage Weil in 2004. The project was then purchased by Red Hat in 2011 and the first stable release of CephFS came out in 2012.

A. Goals

It provides very scalable, object, block file based storage solutions. Ceph has a single storage platform which can handle object, block and file mechanisms. It's very highly scalable with the file system even going upto exabyte level.

B. Architecture

Ceph nodes takes advantage of hardware and intelligent daemons and a ceph storage cluster which accommodates a huge number of nodes, which communicate with each other in a dynamic fashion which redistributes and replicate data. It provides us with its infinitely scalable cluster called RADOS. CephsFS has 2 types of daemons called the ceph monitor and ceph osd monitor. The monitor keeps a record for which only the monitor has root access to. The monitor is in place to ensure that there's always high availability in case a monitor daemon fails. The role of ceph osd monitor is to monitor its own state and the state of other daemons and reports to the monitor.

C. Process

The clients of ceph and osd daemons uses the CRUSH algorithm to effectively compute information about object location instead of looking up in a central master record.

D. Communication

The clients before reading and writing data to storage, according to protocol must contact the monitor which will fetch or refer the most updated version of the cluster monitor.

E. Synchronization

All HDFS communication protocols are layered on top of the TCP/IP protocol. The client can establish a connection with the Name Node or Master node through TCP port which is configurable.

F. Fault Tolerance

For better fault tolerance cephfs supports a cluster of monitors. When there's a cluster of monitors[10] and when a lot of communication takes place there are chances due to latency and other faults can cause it to not update the current state of the cluster. Hence because of this ceph has an agreement between various monitors regarding the current state of the cluster. It uses a paxos algorithm to establish a consensus between the several ceph monitors.

V. CONCLUSION

The DFSs are a very widely used for sharing of of shared permanent storage. In this paper, we have tried to develop a taxonomy and used against existing and renowned distributed file systems like andrew file system, network file system, etc were reviewed and surveyed. Network file system is very popular and one of the most used distributed FS for many days. AFS uses very strong client side caching which is one of the pluses of AFS. But AFS also has some weak points for example it's hard to setup and very complex. For applications which use big streams of data needs to be stored distributed file systems of google and hadoop is considered. Distributed file system provided by google is fully distributed. The down side of GFS is that it is not an open source. HDFS is open source making it available for any user. It provides a plethora of opportunities for massive scale. It is a highly fault tolerant and can also be scaled easily. As HDFS posses the advantages stated above many research is still going on this. The one major drawback of Hadoop is that most of the organizations are modifying its direction for their own needs.

REFERENCES

[1] Sunita Mahajan"Distributed Computing", Oxford University Press

[2] Andrew Tenanbum,"Distributed System", PHI

[3] Tran Doan Thanh, "A Taxonomy and Survey on Distributed File Systems ", Fourth International Conference on Networked Computing and Advanced Information Management.

[4] Russell Sandbaerg, "Design and Implementation or the SUD
Network File system", Sun Microsystems.

[5] Ghemawat, S., Gobioff, H., Leung, S.T., "The Google file system", ACM SIGOPS Operating Systems Review, Volume 37, Issue 5, pp. 29-43,December, 2003.

[6] Konstantin Shvachko," The Hadoop Distributed File System", Yahoo-Inc.com.

[7] The Hadoop Distributed File System http://hadoop.apache.org/core/docs/current/hdfs_design.html

[8] Maltzahn, C., Molina-Estolano, E., Khurana, A., Nelson, A.J., Brandt, S.A. and Weil, S., 2010. Ceph as a scalable alternative to the hadoop distributed file system. *login: The USENIX Magazine*, *35*, pp.38-49.

[9] Sergey, M., 2017. Tiered file system: Optimization of architecture and management algorithms.

[10]Poat, M.D. and Lauret, J., 2017, November. Achieving cost/performance balance ratio using tiered storage caching techniques: A case study with CephFS. In *J. Phys. Conf. Ser.*(Vol. 898, p. 062022).