ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Website:www.ijircce.com

Vol. 5, Issue 2, February 2017

Study of Distributed File System for Big Data

Sagar A. Zalte[1], Vishwas R. Takate[2], Saish R. Chaudhari[3]

Assistant Professor, Department of Electronics and Telecommunication, K.K.W.I.E.E.R, Nashik, India [1]

Assistant Professor, Department of Electronics and Telecommunication, K.K.W.I.E.E.R, Nashik, India [2]

Assistant Professor, Department of Electronics and Telecommunication, K.K.W.I.E.E.R, Nashik, India [3]

ABSTRACT: The caption for Hadoop is big data analytics. That means perform Analytics over big data. Traditional technologies such as R, SQL, and Vertica cannot deal with big data. Hadoop can store unstructured semi structured and structured data. Two core component of Hadoop are 1.HDFS 2.Map Reduce. The Hadoop Distributed File System (HDFS) is designed to store very large data sets reliably, and to stream those data sets at high bandwidth to user applications. Map Reduce is an execution engine of Hadoop, it process the data stored in HDFS in distributed manner.

KEYWORDS: Hadoop, HDFS, distributed file system, Map-Reduce

I. INTRODUCTION

Hadoop is an Apache project; it is developed in Java. Highly recommended for big data/huge amount of data. Hadoop can store any volume any shape of data.HDFS stores the file in block HDFS is inspired from GFS that is google file system which provides high scalability all components are available via the Apache open source license. HDFS is the file system component of Hadoop.HDFS is distributed file system inspired from google file system. The Hadoop Distributed File System (HDFS) is a distributed file system designed to run on commodity hardware. Hadoop Distributed File System is highly scalable. Minimum block size in Hadoop Distributed File System is 64MB. Hadoop Distributed File System can store unlimited data in distributed manner. Map-Reduce are an execution engine of Hadoop. This is divided in two phases 1.Mapper 2.Reducer

1. Mapper: Mapper separates the required key and value from the given input. If the input is flat file mapper read each line as record and separates the required key and value. Mapper will start running wherever there is an instructed data available. Data processed by mapper is called as input split. Number of mapper are equal to number of unique blocks mapper's output get stored on local file system. Mapper produces the output in the form of key and value only. One cannot control the number of mapper.

2. Reducer: Mapper's output is an input to Reducer. Whenever there is a need of grouping, sorting and aggregation reducer starts working on mapper's output Reducer starts when all mapper finishes their work. This might start grouping and sorting but aggregation starts after all mapper finishes their work. One can control the number of reducer using map.reduce.tasks property.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

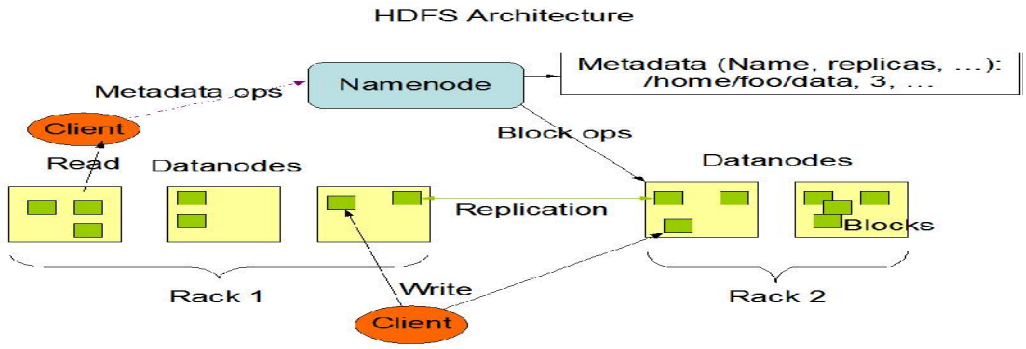
International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Website:www.ijircce.com

Vol. 5, Issue 2, February 2017

II. ARCHITECTURE



A. *NAME NODE :*

HDFS has master/slave architecture. Name node is Master of storage Name node is single point of failure in Hadoop (SPOF). The Name Node executes file system namespace operations like opening, closing, and renaming files and directories. It also determines the mapping of blocks to Data Nodes. HDFS is built using the Java language; any machine that supports Java can run the Name Node. A typical deployment has a dedicated machine that runs only the Name Node software. Each of the other machines in the cluster runs one instance of the Data Node software. The Name Node is the arbitrator and repository for all HDFS metadata. The system is designed in such a way that user data never flows through the Name Node. When storage is requested from client then the request is sent to the Name node name node estimates the required file space in HDFS based on HDFS blocks. It also selects the salves where data would be stored and gives an instruction to the data node. Once file blocks are distributed file blocks metadata will be registered in Name Node.

Name node will maintain other following information: -

1. Configuration (HDFS)
2. Physical address of slaves
3. Space availability info of each slave
4. Maintaining (Edit logs)

Name node is single point of failure if Name node goes down need to replace the new H/W for Name node and Restart the Edit Logs (Edit logs are the file system logs).

B. *DATA NODES :*

Data node is responsible for saving the file blocks and each data node acknowledges the name node also sends heart beat signal to Name node. File is split into one or more blocks and these blocks are stored in a set of Data Nodes. The Data Nodes are responsible for serving read and write requests from the file system's clients. The Data Nodes also perform block creation, deletion, and replication upon instruction from the Name Node Each block replica on a Data Node is represented by two files in the local host's native file system. The first file contains the data itself and the second file is block's metadata including checksums for the block data.

C.JOBTRACKER:

The Job tracker is the service within Hadoop that farms out Map Reduce tasks to specific nodes in the cluster, ideally the nodes that have the data, or at least are in the same rack. Client applications submit jobs to the Job tracker. The Job tracker talks to the Name Node to determine the location of the data. Job tracker process runs on a separate node and not usually on a Data Node. Job tracker is an essential Daemon for Map Reduce execution. When the Job Tracker is down, HDFS will still be functional but the Map Reduce execution cannot be started and the existing Map Reduce jobs will be halted

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Website:www.ijircce.com

Vol. 5, Issue 2, February 2017

Job tracker is responsible for Load balancing during the work. Job tracker manages the task tracker also receives an heart beat signal from task tracker and finds the best Task Tracker nodes to execute tasks based on the data locality.

D. TASK TRACKER:

Task Tracker runs on Data Node. Mostly on all Data Nodes. Task tracker is responsible for Processing the file blocks as per the order of job tracker. Task Tracker processes the data (files). And once complete the process it acknowledges to the job tracker. Task tracker performs the below tasks

1. Mapper Tasks
2. Combiner Tasks
3. Reducer Tasks

E. SECONDARY NAME NODE:

As Name node is single point of failure in Hadoop, Secondary Name Node executes the currently active jobs at the failure point of Name Node. Secondary Name Node contains the metadata of currently executing jobs whereas Name Node contains the metadata for all the files.

Secondary Name Node holds the below information:

1. Temporary Edit Logs.
2. The image of all IP addresses.
3. During the fault tolerance if the Name Node is down then job tracker avails the Meta data from Secondary Name Node
4. Output storage decision is done by Secondary Name Node
5. Meta data of new output file and edit log

III. THE FILE SYSTEM

HDFS supports a traditional hierarchical file organization. A user or an application can create directories and store files inside these directories. The file system namespace hierarchy is similar to most other existing file systems; one can create and remove files, move a file from one directory to another, or rename a file.HDFS does not yet implement user quotas. HDFS does not support hard links or soft links. However, the HDFS architecture does not preclude implementing these features.

The Name Node maintains the file system namespace. Any change to the file system Namespace or its properties is recorded by the Name Node. An application can specify the number of replicas of a file that should be maintained by HDFS. The number of copies of a file is called the replication factor of that file. This information is stored by the Name Node

IV. DATA REPLICATION

HDFS is designed to reliably store very large files across machines in a large cluster. It stores each file as a sequence of blocks; all blocks in a file except the last block are the same size. The blocks of a file are replicated for fault tolerance. The block size and replication factor are configurable per file. An application can specify the number of replicas of a file. The replication factor can be specified at file creation time and can be changed later. Files in HDFS are writing-once and have strictly one writer at any time. The Name Node makes all decisions regarding replication of blocks. It periodically receives Heartbeat and a Block report from each of the Data Nodes in the cluster. Receipt of a Heartbeat implies that the Data Node is functioning properly. A Block report contains a list of all blocks on a Data Node.

V. REPLICA PLACEMENT

The placement of replicas is critical to HDFS reliability and performance. Optimizing replica placement distinguishes HDFS from most other distributed file systems. This is a feature that needs lots of tuning and experience.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Website:www.ijircce.com

Vol. 5, Issue 2, February 2017

The purpose of a rack-aware replica placement policy is to improve data reliability, availability, and network bandwidth utilization. The current Implementation for the replica placement policy is a first effort in this direction. The short-term goals of implementing this policy are to validate it on production systems, learn more about its behaviour, and build a foundation to test and research more sophisticated policies.
Large HDFS instances run on a cluster of computers that commonly spread across many racks. Communication between two nodes in different racks has to go through switches. In most cases, network bandwidth between machines in the same rack is greater than network bandwidth between machines in different racks. Large HDFS instances run on a cluster of computers that commonly spread across many racks. Communication between two nodes in different racks has to go through switches. In most cases, network bandwidth between machines in the same rack is greater than network bandwidth between machines in different racks.

VI. REPLICA SELECTION

To minimize global bandwidth consumption and read latency, HDFS tries to satisfy a read request from a replica that is closest to the reader. If there exists a replica on the same rack as the reader node, then that replica is preferred to satisfy the read request. If HDFS cluster spans multiple data centres, then a replica that is resident in the local data centre is preferred over any remote replica

REFERENCES

1. Apache Hadoop. http://hadoop.apache.org/
2. P. H. Carns, W. B. Ligon III, R. B. Ross, and R. Thakur. “PVFS: A parallel file system for Linux clusters,” in Proc. of 4th Annual Linux Showcase and Conference, 2000, pp. 317–327.
3. J. Dean, S. Ghemawat, “MapReduce: Simplified Data Processing on Large Clusters,” In Proc. of the 6th Symposium on Operating Systems Design and Implementation, San Francisco CA, Dec. 2004.
4. A. Gates, O. Natkovich, S. Chopra, P. Kamath, S. Narayanam, C. Olston, B. Reed, S. Srinivasan, U. Srivastava. “Building a High-Level Dataflow System on top of MapReduce: The Pig Experience,” In Proc. of Very Large Data Bases, vol 2 no. 2, 2009, pp. 1414–1425
5. S. Ghemawat, H. Gobioff, S. Leung. “The Google file system,” In Proc. of ACM Symposium on Operating Systems Principles, Lake George, NY, Oct 2003, pp 29–43.
6. Computing, Calgary, AB, Canada, August 10–12, 2009.

BIOGRAPHY

Sagar A.Zalte, Vishwas R.Takate, Saish R. Chaudhari are Assistant professor in the Electronics and Telecommunication Department, of K.K.WaghCollege of Engineering,Nashik.