International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 08 Issue: 03 | Mar 2021 www.irjet.net p-ISSN: 2395-0072

High Performance Distributed Storage and Distributed Computingusing Hadoop and Containers

Anupran Singh

B.Tech - Computer Science and Engineering, SRM Institute of Science and Technology, Kattankulathur, Chennai, India

---

Abstract - *This paper provides an advanced understanding of the concept Big Data and its components along with an in-depth analysis of the Hadoop framework, a tool used to manage Big Data. This is essentially achieved by showcasing a High-performance tool for Big Data using Hadoop framework and then analyzing various tasks on the cluster i.e., the internal architecture of Hadoop along with resource analysis.This article also embeds ideas that are essential for us to envision prospects in data management techniques. The project also deploys the cluster management process over containerized application using a docker engine. In this article, an in-depth analysis of HDFS and Map Reduce frameworks is followed by a brief showcasing of the contrast between the current deployment of Big data using distributed computing and a rather futuristic version of it using containerized applications.*

***Key Words*: Hadoop, MapReduce, HDFS, Containers and Docker**

1.INTRODUCTION AND SOME RELATED WORK

Big data is essentially a collection of large complex datasets which cannot be processed using our traditional processing techniques in computing. It is not a technology or a tool, it is just data, an ever- e v o l v i n g dataset which is now an essential part almost all engineering domains and industries like health industry, Automobile industry, etc. The Big Data can be most effectively categorized as Structured, Semi-structured and Unstructured data.

1.1 Structured Data

Structured data is basically information extracted from raw data. It includes facts and figures, columns and rows, tables, charts etc. The required information then can be easily extracted using simple data mining tools and techniques. Structured data is used by most organizations effectively.

1.2 Semi-structured Data

Semi-structured data is neither raw data nor conventionally analyzed data. It is a form of structured data which is not yet organized in the form of tables, rows, and columns. Still the data can be extracted effectively. The data found on web can be categorized into semi-structured data.

1.3 Unstructured Data

Unstructured data is the raw form of data and doesn't have any predefined structure or model or schema like any other traditional database technology. It can contain text form without any structure, date and time details, audio, video, images without any reference or meta data.

The six important components of Big Data spectrum are:

1) Volume
2) Velocity
3) Variety
4) Viability
5) Value
6) Veracity

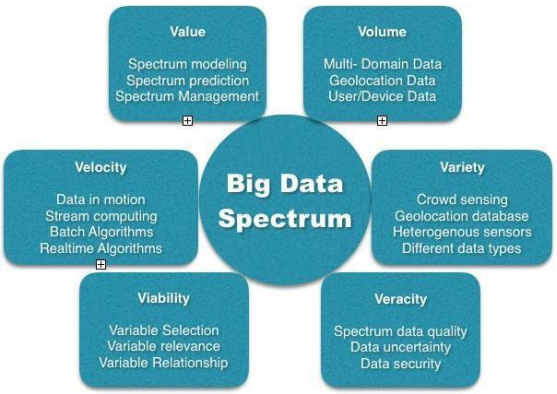

Figure -1: Representation of six V's of Big Data spectrum

1.4 Hadoop

Hadoop is an open-source tool which is used widely to manage big data, it is designed by Apache community. Hadoop provides frameworks that can store and processing large datasets very efficiently. Hence the Hadoop framework embeds distributed file system (for storing data) and distributed computing (for processing data). These frameworks are called **HDFS** (Hadoop Distributed File System) and **Map Reduce.**

1.4.1 HDFS

Hadoop Distributed File System is the technique used for distributed storage. The file system meta data and Application data are stored separate ly in HDFS. Meta data is stored on a dedicated server called as Name Node and the application data is stored on a different server called as Data Nodes. All the servers are connected to each other via TCP protocol. This technique is analogous to as if we formed a large network of hard disks adding up their individual capacities to form a cluster.

1.4.2 MapReduce

This framework is used to process data which is already present in the HDFS cluster. So, MapReduce provides a processing cycle to the Hadoop network. Job Tracker is the master Server which governs other servers and specifies the tasks to other servers. Task Tracker is the slave server and follows the commands of Job Tracker.

1.4.3 Containers

Containers can be defined as portable isolated environments in which several applications can be simultaneously executed. Containers supports all the libraries and dependencies that are required. They can be defined in an informal way, a derivative to virtual machines. Containers share the accessible system resources for computing, networking, and storage process. Containers on one host share the same OS kernel. Container is a technology whereas Docker is a product. Docker instigates the concept of Container images, which allows containers to be used on any host with a kernel. The container images of applications can be deployed rapidly within seconds whereas a virtual machine image takes considerable amount of time to be deployed.

2. HDFS ARCHITECTURE

2.1 Data Node

A Data Node is a slave server in the HDFS cluster, which stores application data and its replicas in the form of data blocks. Initially when the cluster loads up, all the Data Nodes in the cluster establishes a connectionwith the Name Node in the form of a Handshake. This basically verify the namespace ID and software version of the Data Node. If it does not match, the cluster crashes. Every Data Node in the cluster must have the same namespace ID to be a part of the cluster. After the handshake, the Data Node registers itself with the Name Node and generates unique storage ID, after which an HDFS client wanting to store a file in the cluster extracts the metadata and IP address of Data from the Name Node. The client establishes a connection with the IP address given by the Name Node. Then the client divides the file into data blocks.a data block consists of two components, file data and meta data. Length of the data block is equal to the size of file. And then after producing the data block creates replicas of the data block. The Data nodes not responsible for replication but only storage of the data block. When analyzing the cluster at wire level using TCP-dump command, we observe that in the established cluster the Data Nodes send Heartbeats to both Name Node and Client. This process confirms that the Data Node is working properly and finds out if the Data Node hosts and data block. The default heartbeat interval in Hadoop is three seconds but it can be customized accordingly.

2.2 Name Node

HDFS cluster consists of a master node which is called as Name Node. HDFS contains files and directories which. are represented on Name Node containing file permissions, file modifications, access times and disk capacity. This information is called as Meta Data. Then the client establishes a connection with the IP address given by Name node. After successful storage of the data block in the cluster, the Name Node is provided with the meta data of the data block. The process to read a file is also achieved with a similar mechanism. HDFS client provides the name node with the filename and requests to provide the IP address of the data node where that file is stored. This working model of Name Node is hence established by analyzing the connection between different components of the cluster i.e., using a TCP-dump command. During a reading operation, If a connection failureoccurs then the Name Node automatically connects the client to other Data Node with the replica of the data block.

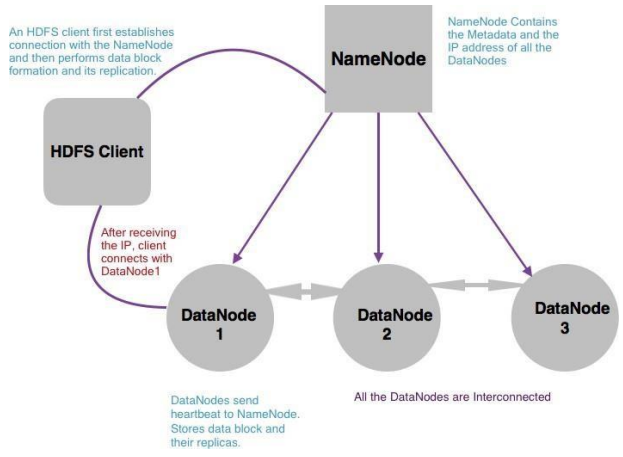

Figure-2: The working model of an HDFS cluster showcasing HDFS Client, Name Node and Data Node.

2.3 HDFS Client

User acts as HDFS Clients to access the Hadoop file system. HDFS only supports read, write and delete operation. HDFS itself as a file system is not equipped for writing operation. To read a file, HDFS Client asks the Name Node to produce the IP address of Data Node which hosts the required data block. The replica generation operation is performed by HDFS Client, not Data Node. In the storage process client is responsible for selection of Data Node for data block.

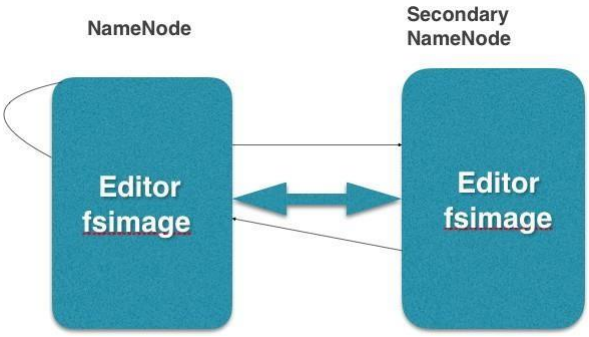

Figure-3: The working model represents the secondary Name Node and Checkpointing.

2.4 Secondary Name Node and I/O operations:

The name node basically stores the HDFS filesystem information in a file called as fsimage. Any updates to the file system are made in the fsimage file, but instead are logged into a file, so the I/O is fast append only streaming as opposed to a random file write. When restarting the name node, it reads the fsimage and then applies all the changes from the log file to bring the filesystem state up to date in memory. This process takes time. So secondary name node is not a backup to the name node, but it read all the filesystem changes log and updates the fsimage file. This reduces the starting time a Name Node takes. And Secondary Name Node fails to provide high availability to the cluster. This is also called as checkpointing. The editor file and fsimage file are both stored in the RAM and not the internal hard disks. We can take a copy of fsimage in case the name node shuts down.

3. HDFS ALGORITHM AND IMPLEMENTATION

HDFS implementation includes configuration of two essential components of HDFS cluster i.e., *Data Node* and *Name Node.*

3.1 Data Node:

A Data Node is created with two files, hdfs-site.xml and core-site.xml.

hdfs-site.xml configuration :

```
cd / etc / hadoop
vim hdfs -site .xml
<configuration>
<property>
<name>dfs . name . dir </name>
<value >/dn </ value >
</ property >
</ configuration >
```

core-site.xml configuration :

```
cd / etc / hadoop
vim core -site .xml
<configuration>
<property>
<name>fs . default . name </name>
<value >hdfs : / / dn:9001 </ value >
</ property >
</ configuration >
```

3.2 Name Node:

A Name Node is also created with these two files hdfs-site.xml and core-site.xml but with slight modifications in the configuration of hdfs-site.xml file.

hdfs-site.xml configuration :

```
cd / etc / hadoop
vim hdfs -site .xml
<configuration>
<property>
<name>dfs . name . dir </name>
<value >/nn </ value >
</ property >
</ configuration >
```

core-site.xml configuration :

```
cd / etc / hadoop
vim core -site .xml
<configuration>
<property>
<name>fs . default . name </name>
<value >hdfs : / / nn :9001 </ value >
</ property >
</ configuration >
```

These two Commands basically start the hadoop cluster.

```
hadoop-daemon . sh start datanode
hadoop-daemon. sh start namenode
```

4. MAP REDUCE ARCHITECTURE

MapReduce is essentially the distributed computing paradigm of Hadoop. It provides processing of the data present in the HDFS cluster. MapReduce processing model is basically divided into two major tasks, Mapping task and the Reducing task.

4.1 Job Client:

Job Client is the interface which interacts with the cluster. So, a Job Client is able to submit jobs, track the progress of jobs, and has status information for MapReduce cluster. A user application can act as a Client in order to get sometasks completed by the MapReduce Framework. Same client. can be a part of HDFS as well as MapReduce. The client maintains a connection with the Job Tracker as well as the Name Node. A client wanting to process a file approaches the Job Tracker with the details of the file i.e., file name and the program it wants to execute on that file.

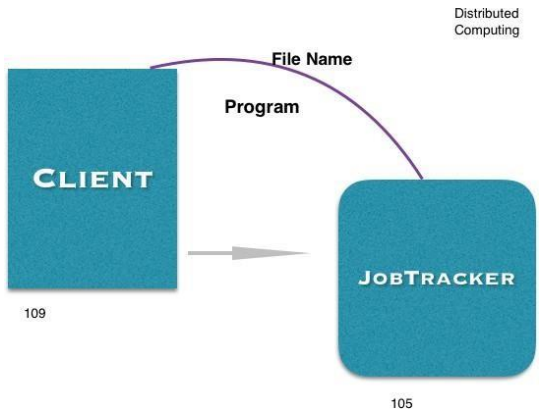

Figure-4: The Client approaches the Job Tracker withthe filename andprogram to be executed on the file.

4.2 Job Tracker:

A JobTracker is the master server in the MapReduce cluster. The most important functionality of the JobTracker is to manage resources in the cluster. It takes in accountthe responsibility to manage all TaskTrackers (Resource Management) including avalibility of the resources, tracking progress and faults. JobTracker accepts the request from the Client in the form of a job and the assigns TaskTracker with the task to be performed in the process.

4.3 Task Tracker:

A Task Tracker is the slave server in the MapReduce cluster.It executes the Map and Reduce operations by accepting the orders from the JobTracker in the form of a Job. The TaskTracker just follows the instructions given by JobTracker and simultaneously updated the JobTracker with the progress in the task. TaskTracker accepts the request from theClient in the form of a job and the assigns TaskTracker with the task to be performed in the process. A TaskTracker has a fixednumber of slots on which it can take a job , hence during the scheduling of the job ,the JobTracker starts looking for an empty slot in the TaskTracker on the same server as of theDataNode which contains the data for that task.

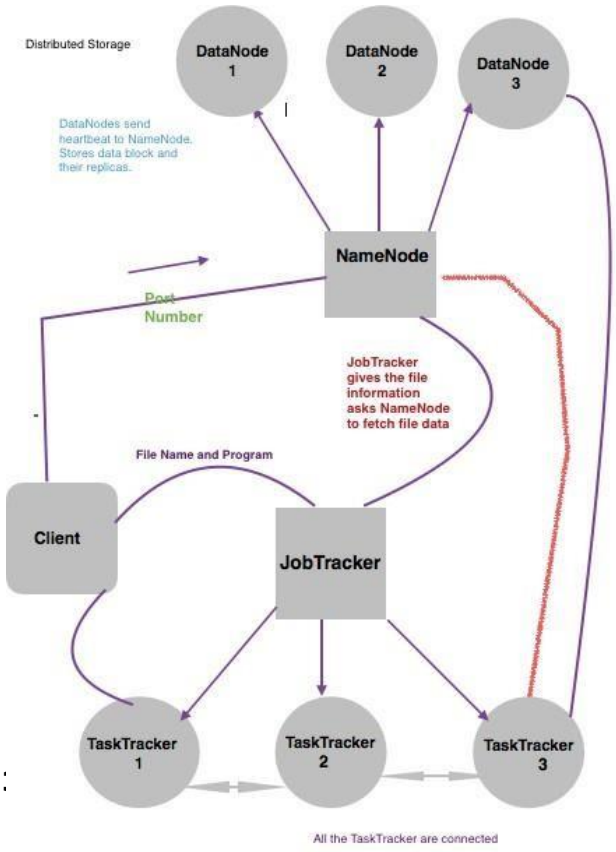

Figure-5: A MapReduce architecture to showcase the Functioning of the Process model. The Data Node and Task Tracker are hosted by different servers.

4.4 Mapper and Reducer:

When the Job Client Approaches a Job Tracker to run a job, it basically provides the Job Tracker with two information, the filename on which the job is to run and the programs for to execute on the file during the process. These programs are the core to processing in the MapReduce operations. Now after receiving the data from HDFS cluster the file is dividedinto ten equal parts called as Input Split. Now some Task Trackers are specified with mapping task and some with reducing task by the Job Tracker randomly. A Task Tracker by default is capable to handle only two mapper operation and two reducer operation in the batch process. Mapper is a first phase into solving a problem. Mapper program is the logical approach. It includes computation, processing, and distribution of data. Reducerfollows up the Mapping process, and it is the last phase todesign the solution.

5. WORKING MODEL

- Considering an already prepared HDFS cluster, the Job Client approaches the Job Tracker to run a Job and provides filename, mapper and reducer programs tothe Job Tracker.
- It is believed that the Job Tracker which is connected to the Name Node in the HDFS cluster approaches Name Node to fetch information about the file stored in data node. But it was observed thatthe Job Tracker asks One of the Task Trackers to fetchinformation from the Name Node.
- During the operation on the wire level, using tcp-dump command, we observe a connection was established between the Task Tracker and Name Node. So the red line in the architecture proves this hypothesis.
- Now the Name Node fetches the file information (On which Data Node, it is stored) to the Task Tracker. Task Tracker then establishes a connection with thatparticular Data Node and loads up the file data.
- Now this data chunk is divided into several equal parts called as Input Splits.

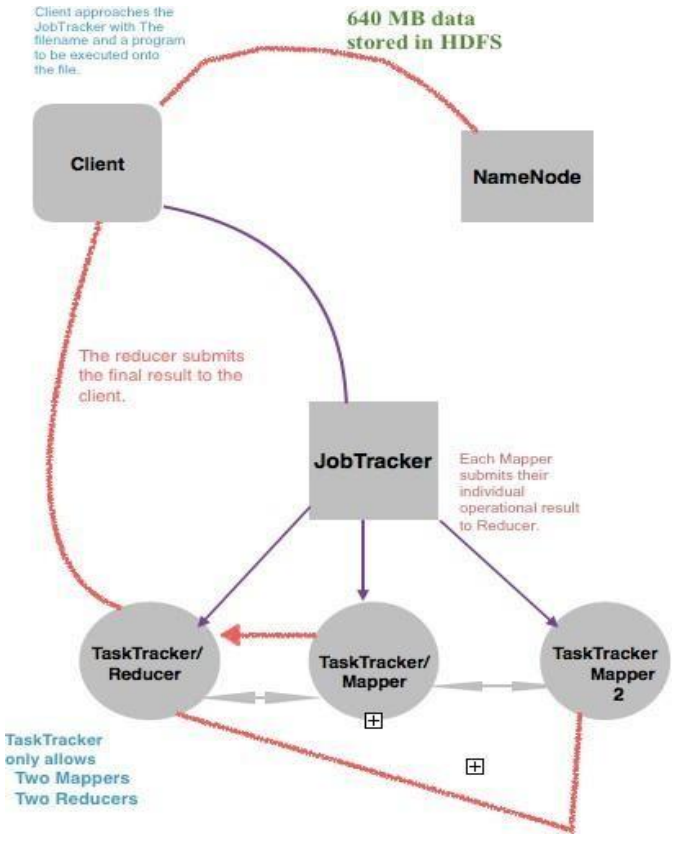

Figure-6: Working of a Task Tracker as a Mapper/Reducer

- To execute word count for a piece of document which is six pages long, consider that the Job Tracker distributes these six pages to six individual Task Trackers and these Task Trackers are now subjected to mapping process and are now called mappers. Whereas these mappers after word count submits their individual results to the Reducer(another Task Tracker employed by the JobTracker to perform reducing operation) . The reducer then runs the final operation to obtain the final result. This final result is submitted by the reducer to the Job Client.
- It was also observed using tcp-dump command that at wire level the Reducer establishes a connection with Job Client hence this proves that the result after a job is executed, is submitted to Client by Reducer(essentially a TaskTracker) and not a JobTracker.

6. MAP-REDUCE ALGORITHM IMPLEMENTATION

MapReduce implementation includes certain factors such as configuration of some essential components of processing models i.e. *Job client* , *Job Tracker* and *Task Tracker* followed by Job analysis then mapping and reducing process.

6.1 Job Client:

A client is the external user application in the MapReduce process. It is configured by two files i.e core-site.xml and mapred-site.xml.

mapred-site.xml configuration:

```
cd / etc / hadoop
vim mapred-site.xml
<configuration>
<property>
<name>mapred.job.tracker</name>
<value>jt:9000</value>
</property>
</configuration>
```

core-site.xml configuration:

```
cd / etc / hadoop
vim core-site.
xml
<configuration>
<property>
<name>fs.default.name</name>
<value>hdfs://nn:9001</value>
</property>
</configuration>
```

6.2 Job Tracker:

Install JDK and Hadoop

mapred-site.xml configuration:

```
cd / etc / hadoop
vim mapred-site.xml
<configuration>
<property>
<name>mapred.job.tracker</name>
<value>jt:9002</value>
</property>
</configuration>
```

core-site.xml configuration:

```
cd / etc / hadoop
vim core-site.xml
<configuration>
<property>
<name>fs.default.name</name>
<value>hdfs://nn:9001</value>
</property>
</configuration>
```

6.3 Task Tracker:

Check and Install JDK and Hadoop first.

mapred-site.xml configuration:

```
cd /etc/hadoop
vim mapred-site.xml
<configuration>
<property>
<name>mapred.job.tracker</name>
<value>tt:9002</value>
</property>
</configuration>
```

core-site.xml configuration:

```
cd/etc/hadoop
vim core-site.xml
<configuration>
<property>
<name>fs.default.name</name>
<value>hdfs://nn:9001</value>
</property>
</configuration>
```

Commands:

```
hadoop-daemon.sh start jobtracker
*To start JobTracker*

hadoop-daemon.sh start tasktracker
*To start TaskTracker

/user/java/jdk.../bin/jps
*To check the status of current Job*

hadoop job-list-active-tasktrackers
*this command lists out all activeTask Trackers.*

hadoop dfsadmin-report | grep.name
*GREP command gives the report and listsout
all the IP's.*
```

7. CONTAINERS

The inception of the an ON-DEMAND service in our project was driven by this revolutionizing technology called Containers. Containers are the isolated portable instances in which we can run applications. A virtual machine takes about twenty minutes to install and boot an operating system. In contrast to this when a container performs the same operation itcan be executed in just a couple of seconds. And the operating system generated within a container is a full flagged Operating System with its own IP address, network card and Libraries. Considering only the Base Operating System, the resources in a system are wasted because the Operating System only uses about five percent of the CPU storage. Another drawback observed was the duplication of code and installation of the same Operating System multiple times As evident in the architectures of Virtual machine and Container we observe, the virtual OS's created in Virtual machine share only hardware resources from the base system. A Hypervisor layer is used above the host Operating System.

Whereas in containers, the container image generated is capable of sharing both the hardware resources as well as the software resources in the system. The hypervisor layer is replaced with a docker engine. The operating systems are formed in container images, having a full flagged working Operating System. These images share RAM, Kernel and other resources with the base Operating System , but have different Operating System libraries with each having unique MAC address ,network card and IP address. This technology is called as Container Technology and the product originated from this technology is called as Docker. Container technology is implemented using Docker. Docker increases the possibility to explore in the Big Data field. Since we can launch thousands of Operating Systems in matter of seconds, part of our project focuses on profoundness of Docker. By introducing a feature called as ON DEMAND service we essentially take our virtual machine paradigm on Container services and reinvent our cluster to provide a rather efficient and faster model.

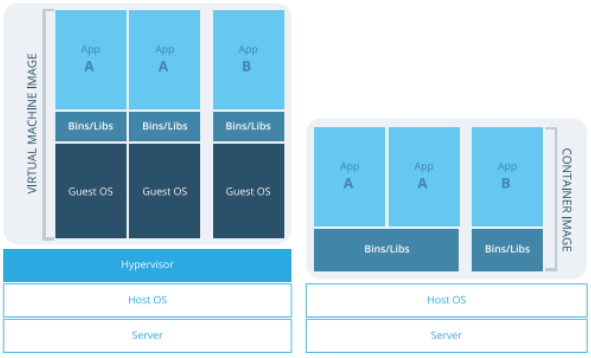

Figure-7: Virtual Machine vs. Container

8. IMPLEMENTING CONTAINERS USING DOCKER

Container is a technology, and it is implemented using Dockers.

1. Installation of docker engine.
   docker engine 1.9.1 1.wl7.centos.x86 64.rpm

   yum install docker–io

2 Displays number of docker imagest h a t areworking.
   docker images / docker ps

3 Genration of docker image of anOS.
   docker pull OS name

4. Retriving the previous datainthe image.
   docker start [Container ID]
5. Tolaunch MulltipleDockers.

```
for(i=0;i<10;i++)

>do
>docker run i t d name=
project$iredhat
```

[6]
[7]

6. Deleting all images at a time.docker

9. CONCLUSIONS

Hadoop framework is used to store structured as well as unstructured data like images, video etc. HDFS (Hadoop Distributed File system) is used to store data in an HDFS cluster. MapReduce processing model can be used to process data already stored in the Hadoop cluster. HDFS does not provide parallel storage in the HDFS cluster and thus it takes more time to store data in an HDFS cluster than If the data was stored individually in the Data Nodes. HDFS allows read and delete operations to a file, but once a file is uploaded in the cluster it cannot be written again,so HDFS does not allow append operation.

MapReduce model provides parallel processing of data but does not allow real time processing. It means the data needs to be stored first in the HDFS cluster only then it can be processed, this is known as Batch Processing. MapReduce only offers Batch processing whereas Spark offers real time processing. Containerization and Docker is a perfect realization of what we lack in terms of efficiency, when dealing with Big Data using Tools. It can be a replacement to Virtual Machines.

ACKNOWLEDGEMENT

This research paper is the result of insightful wisdom given to us by several professors whose research work I referenced to from time to time. I would like to show my gratitude towards the open stack society for their support and documentation is the key to my enthusiasm and ability to work really hard on such a niche subject as Docker. This paper is sincerely dedicated to all the authors, who's profound research helped me to write this paper.

REFERENCES

[1] Apache Hadoop. http://hadoop.apache.org/
[2] 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.
[3] OpenStack, "Exploring Opportunities: Containers and OpenStack".www.openstack.org
[4] "HADOOP-6330: Integrating IBM General Parallel File System im- plementation of Hadoop File System Interface". IBM. 2009-10-23.
[5] Under the Hood: Scheduling mapReduce jobs more efficiently with Corona". Facebook. Retrived 2012-11-9.
www.stackoverflow.com
"HDFS Architecture". Retrived 1 September 2013.
[8] M. Schatz, Blastreduce: high performance short read mapping with mapreduce, University of Maryland, http://cgis.cs.umd.edu/Grad/scholarlypapers/papers/MichaelSchatz.pdf
[9] A. Bialecki, M. Cafarella, D. Cutting, and O. OMALLEY, Hadoop: a framework for running applications on large clusters built of commod-ity hardware, Wiki at http://lucene.apache.org/hadoop, vol. 11, 2005.
[10] S. Chen, B. Mulgrew, and P. M. Grant, A clustering technique for digital communications channel equalization using radial basis function networks, IEEE Trans. Neural Networks, vol. 4, pp. 570578, July 1993.

BIOGRAPHIES

My name is Anupran Singh, and I am a Computer Engineer successful at working with teams, driving progress toward project milestones, quality assurance, and on-time delivery. Offering 2+ years of experience in software development. Working towards improving my Knowledge base and analysis skills. Looking to make a dent on my learning curve.