Hadoop Architecture and HDFS

Homework 2

Q1. What are HDFS and YARN?

The Hadoop Distributed File System (HDFS) is a distributed file system designed to run on commodity hardware. HDFS provides high throughput access to application data and is suitable for applications that have large data sets.

The fundamental idea of YARN is to split up the functionalities of resource management and job scheduling/monitoring into separate daemons. The idea is to have a global ResourceManager (*RM*) and per-application ApplicationMaster (*AM*). An application is either a single job or a DAG of jobs.

Q2. What are the various Hadoop daemons and their roles in a Hadoop cluster?

Apache Hadoop consists of the following Daemons:

- NameNode
- DataNode
- Secondary Name Node
- Resource Manager
- Node Manager

1. **NameNode** - The primary purpose of Namenode is to manage all the MetaData.

2. **DataNode** - The NameNode always instructs DataNode for storing the Data.

3. **Secondary NameNode** - Secondary NameNode is used for taking the hourly backup of the data.

4. **Resource Manager** - The Resource Manager Manages the resources for the applications that are running in a Hadoop Cluster.

5. **Node Manager** - The Node Manager works on the Slaves System that manages the memory resource within the Node and Memory Disk.

Q3. Why does one remove or add nodes in a Hadoop cluster frequently?

In a Hadoop cluster a Manager node will be deployed on a reliable hardware with high configurations, the Slave node will be deployed on commodity hardware. So the chance of data nodes crashing is more . So more frequently you will see the admin remove and add new data nodes in a cluster.

Q4. What happens when two clients try to access the same file in the HDFS?

Multiple clients can write into HDFS files at the same time. When a client is granted permission to write data on a data node block, the block gets locked till the completion of a write operation. If some other client requests to write on the same block of the same file then it is not permitted to do so.

Q5. How does NameNode tackle DataNode failures?

Data blocks on the failed Datanode are replicated on other Data Nodes based on the specified replication factor in hdfs-site xml file. Once the failed data nodes come back the Name node will manage the replication factor again. This is how Namenode handles the failure of data nodes.

Q6. What will you do when NameNode is down?

When the NameNode goes down, the file system goes offline. There is an optional SecondaryNameNode that can be hosted on a separate machine. It only creates checkpoints of the namespace by merging the edits file into the fsimage file and does not provide any real redundancy.

Q7. How is HDFS fault tolerant?

The HDFS is highly fault-tolerant that if any machine fails, the other machine containing the copy of that data automatically becomes active. Distributed data storage - This is one of the most important features of HDFS that makes Hadoop very powerful. Here, data is divided into multiple blocks and stored into nodes

Q8. Why do we use HDFS for applications having large data sets and not when there are a lot of small files?

HDFS is more efficient for a large number of data sets, maintained in a single file as compared to the small chunks of data stored in multiple files.

Q9. How do you define "block" in HDFS? What is the default block size in Hadoop 1 and in Hadoop 2? Can it be changed?

a)Blocks are the smallest continuous location on your hard drive where data is stored.HDFS stores each file as blocks, and distributes it across the Hadoop cluster.
b)The default size of a block in HDFS is 128 MB (Hadoop 2.x) and 64 MB (Hadoop 1.x)
c)Yes we can change the Hadoop block size.