Hadoop - Namenode, DataNode, Job Tracker and TaskTracker

Namenode

Namenode is the node which stores the filesystem metadata i.e. which file maps to what block locations and which blocks are stored on which datanode.

The namenode maintains two in-memory tables, one which maps the blocks to datanodes (one block maps to 3 datanodes for a replication value of 3) and a datanode to block number mapping. Whenever a datanode reports a disk corruption of a particular block, the first table gets updated and whenever a datanode is detected to be dead (because of a node/network failure) both the tables get updated.

Failover semantics: The secondary namenode regularly connects to the primary namenode and keeps snapshotting the filesystem metadata into local/remote storage.

DataNode

The data node is where the actual data resides.

Some interesting traits of the same are as follows:

- All datanodes send a heartbeat message to the namenode every 3 seconds to say that they are alive. If the namenode does not receive a heartbeat from a particular data node for 10 minutes, then it considers that data node to be dead/out of service and initiates replication of blocks which were hosted on that data node to be hosted on some other data node.
- The data nodes can talk to each other to rebalance data, move and copy data around and keep the replication high.
- When the datanode stores a block of information, it maintains a checksum for it as well. The data nodes update the namenode with the block information periodically and before updating verify the checksums. If the checksum is incorrect for a particular block i.e. there is a ***disk level corruption for that block,*** it skips that block while reporting the block information to the namenode. In this way, namenode is aware of the disk level corruption on that datanode and takes steps accordingly.

Job Tracker and TaskTracker

- The primary function of the job tracker is resource management (managing the task trackers), tracking resource availability and task life cycle management (tracking its progress, fault tolerance etc.)
- The task tracker has a simple function of following the orders of the job tracker and updating the job tracker with its progress status periodically.
- The task tracker is pre-configured with a number of slots indicating the number of tasks it can accept. When the job tracker tries to schedule a task, ***it looks for an empty slot in the tasktracker running on the same server which hosts the datanode where the data for that task resides. If not found, it looks for the machine in the same rack***. There is no consideration of system load during this allocation.
- ***HDFS is rack aware*** in the sense that the namenode and the job tracker obtain a list of rack ids corresponding to each of the slave nodes (data nodes) and creates a mapping between the IP address and the rack id. HDFS uses this knowledge to replicate data across different racks so that data is not lost in the event of a complete rack power outage or switch failure.
- ***Job Performance*** - Hadoop does speculative execution where if a machine is slow in the cluster and the map/reduce tasks running on this machine are holding on to the entire map/reduce phase, then it runs redundant jobs on other machines to process the same task, and whichever task gets completed first reports back to the job tracker and results from the same are carried forward into the next phase.
- ***Fault Tolerance -***
    - The task tracker spawns different JVM processes to ensure that process failures do not bring down the task tracker.
    - The task tracker keeps sending heartbeat messages to the job tracker to say that it is alive and to keep it updated with the number of empty slots available for running more tasks.
    - From version 0.21 of Hadoop, the job tracker does some checkpointing of its work in the filesystem. Whenever, it starts up it checks what was it upto till the last CP and resumes any incomplete jobs. Earlier, if the job tracker went down, all the active job information used to get lost.
- The status and information about the job tracker and the task tracker are exposed vis jetty onto a web interface.