Haddop DFS: Critique

Abhishek Srivastava
Student Id: 861307778

CS202 Advanced Operating Systems
February 27, 2017

Summary

The paper tries to solve the issue of providing a distributed file system(HDFS) which is capable of storing very large amounts of data across multiple nodes which can be easily scalable, can be accessed at very high speed and provides high reliability as well.

Main Points

HDFS is a file system component of Hadoop whose interface is similar to UNIX file system but functionality is changed to improve the performance for the user applications. HDFS stores file system metadata and application data separately. Dedicated Namenode server is used to store metadata. Datanodes servers are used to store the application data which is split into multiple blocks. These servers are connected and use TCP protocols to communicate among themselves. HDFS uses data replication of blocks over different Datanodes to provide data reliability.

HDFS architecture consist of following components:

- Name Nodes: It maintains *namespace tree* which is a hierarchy of directories and files i.e *inodes* information. It also stores mapping information of file blocks to corresponding Data Nodes. It can also stores *checkpoints* and *journal* as well.
- Data Nodes: It stores data blocks using two files, Data file and its metadata which includes generation stamp and checksums of the block. It also stores *namespace ID* and *software version* which is used during connection setup. It also sends *heartbeats* on regular intervals to confirm its and its stored blocks availability.
- HDFS Client: It provides the interface used by applications to access the file system. APIs provided by it can be used by frameworks and admins to configure HDFS.
- Image and Journal: Images can be used to create checkpoints using which we can handle corrupt or lost information situations. Journal is used to handle write bottleneck of multithreaded system by performing batch transactions.
- Checkpoint, Backup Node and File System Snapshots: Provides additional functionality to provide reliability of stored data and nodes recovery.

To read a file, HDFS client asks NameNode information of all the DataNodes which have its data blocks, then it sends request to the closest DataNode for transfer of desired data block. To write data, clients asks NameNode to nominate DataNode where data can be written using its stored metadata information. Client then creates pipeline to write the data blocks.

Multiple optimization are performed for File I/O Operations, Block Placement, Replica Management, Decommissioning Nodes and Inter-Cluster Data Copy.

Limitations/Critique

The Drawbacks of HDFS can be high dependency on NameNodes whose failure can affect efficiency of HDFS in significant way. Unable to modify files after it is written and Expensiveness for small data operations can also be limitations of HDFS.

Proposed Extension

NameNodes can be scaled to handle unresponsiveness when most of the memory is used. Applications or Users centric namespaces can be used to increase performance rather than Cluster centric namespaces.