**Assignment 3.5**

**Problem Statement 1:**

**Explain what High availability of Name node:**

In Hadoop cluster HDFS is used to store big data. In HDFS NameNode is a master daemon.The namenode holds all the metadata about the big data stored in Hadoop file system. In Hadoop cluster if it has a single NameNode, and the particular machine or process became unavailable, the entire cluster would be unavailable until the NameNode was either restarted or started on a separate machine. When namenode fails to perform then in order to ensure high availability of data standby node becomes active. The standby node then starts performing the job of namenode.

The Standby NameNode maintains enough state to provide a fast failover. In order for the Standby node to keep its state synchronized with the Active node, both nodes communicate through a group of separate daemons called JournalNodes. The file system journal logged by the Active NameNode at the JournalNodes is consumed by the Standby NameNode to keep its file system namespace in sync with the Active.

The ZooKeeper Failover Controller (ZKFC) is responsible for HA monitoring of the NameNode service and for automatic failover when the Active NameNode is unavailable. There are two ZKFC processes – one on each NameNode machine. ZKFC uses the Zookeeper Service for coordination in determining which is the Active NameNode and in determining when to failover to the Standby NameNode.

The ZooKeeper Failover Controller (ZKFC) is responsible for HA monitoring of the NameNode service and for automatic failover when the Active NameNode is unavailable. There are two ZKFC processes – one on each NameNode machine. ZKFC uses the Zookeeper Service for coordination in determining which is the Active NameNode and in determining when to failover to the Standby NameNode. In this way the high availability of namenode is ensured.

**Problem Statement 2:**

**Explain what is check pointing and how it is useful:**

Checkpointing is very important in order to maintain file system metadata in HDFS. It’s required for efficient NameNode recovery and restart. The responsibility of NameNode is storing the HDFS namespace. This includes directory tree, file permissions and the mapping of files to block IDs.

A Checkpoint Node was introduced to solve the drawbacks of the NameNode. The changes are just written to edits and not merged to fs image during the runtime. If the NameNode runs for a while edits gets huge and the next startup will take even longer because more changes have to be applied to the state to determine the last state of the metadata.

The Checkpoint Node fetches periodically fsimage and edits from the NameNode and merges them. The resulting state is called checkpoint. After this is uploads the result to the NameNode.

It is important that this metadata (and all changes to it) are safe for stable storage. The file system metadata is stored in two different constructs: the fsimage and the edit log. The fsimage is a file that represents a point-in-time snapshot of the file system’s metadata. Checkpointing is a process that takes an fsimage and edit log and compacts them into a new fsimage. This way, instead of replaying a potentially unbounded edit log, the NameNode can load the final in-memory state directly from the fsimage. This is a far more efficient operation and reduces NameNode startup time.

**Problem Statement 3:**

**Explain HDFS federation:**

The solution to expanding Hadoop clusters indefinitely is to federate the NameNode. Before Hadoop 2 entered the scene, Hadoop clusters had to live with the fact that NameNode placed limits on the degree to which they could scale. Few clusters were able to scale beyond 3,000 or 4,000 nodes.

A Block Pool is a set of blocks that belong to a single namespace. Datanodes store blocks for all the block pools in the cluster. It is managed independently of other block pools. This allows a namespace to generate Block IDs for new blocks without the need for coordination with the other namespaces. The failure of a namenode does not prevent the datanode from serving other namenodes in the cluster.

Federation adds namespace horizontal scaling. Large deployments or deployments using lot of small files benefit from namespace scaling by allowing more Namenodes to be added to the cluster. File system throughput is not limited by a single Namenode. Adding more Namenodes to the cluster scales the file system read/write throughput. A single Namenode offers no isolation in a multi user environment. For example, an experimental application can overload the Namenode and slow down production critical applications. By using multiple Namenodes, different categories of applications and users can be isolated to different namespaces.

**Problem Statement 4:**

**What are the configuration files that are to be edited for sure while installing a Hadoop cluster:**

While installing Hadoop cluster there are four files that need to be configured explicitly while setting up a single node Hadoop cluster as below:

* **Core-site.xml** Configuring the name node address Configuring the rack awareness factor selecting the type of security.
* **HDFS-site.xml** Configure port access Manages ssl client authentication

Controls Network interface Changes file permission.

* **YARN-site.xml** WebAppProxy Configuration MapReduce Configuration NodeManager Configuration ResourceManager Configuration IPC Configuration.
* **Xml**

When Hadoop runs for any analysis of dataset, the framework at runtime for MapReduce jobs is a vast set of rules for assigning jobs to slave and maintain the jobs records. Here YARN in Hadoop2.x is introduced to help this framework to work efficiently and take the workload for job related assignments. It is again a large unit of Hadoop ecosystem which helps running the map and reduce the collaboration with YARN.

**Some of the important features it handles are:**

Node health script variables

Proxy Configuration

Job Notification Configuration