1.

High availability of Namenode

The concept of High Availability cluster was introduced in Hadoop 2.x to solve the single point of failure problem in Hadoop 1.x. In 1.x , introduction of Secondary NameNode did prevent us from data loss and offloading some of the burden of the NameNode but, it did not solve the availability issue of the NameNode. The NameNode becomes a **single point of failure**. It happens because the moment the NameNode becomes unavailable, the whole cluster becomes unavailable until someone restarts the NameNode or brings a new one.

The reasons for unavailability of NameNode can be:

1. for maintenance

2. if namenode crashes

To configure high availability,we are allowing it to have two NameNodes in an active/passive configuration. So, we have two running NameNodes at the same time in a High Availability cluster:

* Active NameNode
* Passive NameNode.

If active node fails , passive node takes the responsibility of active node which is also connected to all

Datanodes and it will happen automatically. With this, we can have automatic failover whenever a NameNode crashes (unplanned event) or we can have a graceful (manually initiated) failover during the maintenance period.

It is bone in two methods

1.Using journal nodes

2. Shared Storage using NFS

1.

The standby NameNode and the active NameNode keep in sync with each other through a separate group of nodes or daemons -called **JournalNodes**. The JournalNodes follows the ring topology where the nodes are connected to each other to form a ring. The JournalNode serves the request coming to it and copies the information into other nodes in the ring.This provides fault tolerance in case of JournalNode failure. The active NameNode is responsible for updating the EditLogs (metadata information) present in the JournalNodes.

2.

* The StandbyNode and the active NameNode keep in sync with each other by using a shared storage device. The active NameNode logs the record of any modification done in its namespace to an EditLog present in this shared storage. The StandbyNode reads the changes made to the EditLogs in this shared storage and applies it to its own namespace.

2.

check pointing

After every restart of namenode , its editlog merges with the fsimage . fsimage has total image of HDFS .it is happened automatically. if we want to do manually ,we use checkpointing concept for special

reasons like if we do fsimage updates regularly ,we have fresh metadata for processing. It is done secondary node to decrease the namenode load . After merging in the secondary namenode ,new fs image is sent back to name node. Checkpointing interval is set manually based on our requriment,generally it will be 1 hr.

3.

Hadoop Federation:

In Hadoop 1.x , no of nodes are limited due to single namenode and its fixed memory but in Hadoop 2.x

Hadoop has introduced HDFS Federation We can scale up the no of namenodes horizontally.

 Following are the main benefits of HDFS Federation:

1. Namespace scalability

HDFS cluster storage scales horizontally, but the namespace does not.

1. Performance

 Adding more NameNodes to the cluster scales the filesystem read/write operation’s performance.

1. Isolation

With multiple NameNodes, different categories of applications and users can be isolated to different namespaces.

4.

Configuration files to be edited for installation of Hadoop:

1. Core-site.xml

2. HDFS-site.xml

3. YARN-site.xml

4. xml