**High availability of Namenode**

* Hadoop 2.x overcomes its Single Point Of Failure shortcoming by providing support for multiple NameNodes. It introduces Hadoop 2.x High Availability feature that brings in an extra NameNode (Passive Standby NameNode) to the Hadoop Architecture which is configured for automatic failover.
* The main motive of the Hadoop 2.x High Availability project is to render availability to big data applications 24/7 by deploying 2  Hadoop NameNodes –One in active configuration and the other is the Standby Node in passive configuration.
* Earlier there was one Hadoop NameNode for maintaining the tree hierarchy of the HDFS files and tracking the data storage in the cluster. Hadoop 2.x High Availability allows users to configure Hadoop clusters with uncalled- for NameNodes so as to eliminate the probability of SPOF in a given Hadoop cluster. The Hadoop Configuration capability allows users to build clusters horizontally with several NameNodes which can operate autonomously through a common data storage pool, thereby, offering better computing scalability when compared to Hadoop 1.x
* With Hadoop 2.x, Hadoop architecture is now configured in a manner that it supports automated failover with complete stack resiliency and a hot Standby NameNode.

**Check pointing and how it is useful**

Hadoop Distributed File System (HDFS) is a journaled file system, where new changes to files in HDFS are captured in an edit log that’s stored on the NameNode in a file named. Periodically, when the file reaches a certain threshold or after a certain period has elapsed, the journaled entries need to be committed to the master file. Checkpointing services for a Hadoop cluster are handled by one of four possible daemons, which need to run on their own dedicated master node alongside the NameNode daemon’s master node:

* Secondary NameNode
* Checkpoint Node
* Backup Node
* Standby NameNode

When it’s time to perform the checkpoint, the NameNode creates a new file to accept the journaled file system changes. It names the new file. As a result, the file accepts no further changes and is copied to the checkpointing service, along with the file. The checkpointing service merges these two files, creating a file named.The checkpointing service copies the file to the NameNode.It overwrites the file and renames is too.

**HDFS federation**

Hadoop federation separates the namespace layer and storage layer. It enables the block storage layer. It also expands the architecture of an existing HDFS cluster to allow new implementations and use cases. The current HDFS architecture has two layers –

Namespace – This layer manages files, directories and blocks. This layer supports the basic file system operations e.g. listing of files, creation of files, modification of files and deletion of files and folders.

Block Storage – This layer has two parts –

Block Management This manages the datanodes in the cluster and provides operations like creation, deletion, modification and search. It also takes care of the replication management.

Physical Storage This stores the blocks and provides access for read or write operations.

**Hadoop federation** allows scaling the name service horizontally. It uses several namenodes or namespaces which are independent of each other. These independent namenodes are federated i.e. they don’t require inter coordination. These datanodes are used as common storage by all the namenodes. Each datanode is registered with all the namenodes in the cluster. These datanodes send periodic reports and responds to the commands from the name nodes. We have a block pool which is a set of blocks that belong to a single namespace. In a cluster, the datanodes stores blocks for all the block pools. Each block pool is managed independently. This enables the name space to generate block ids for new blocks without informing other namespaces. If one namenode fails for any reason, the datanode keeps on serving from other namenodes.

One namespace and its block are collectively called Namespace Volume. When a namespace or a namenode is deleted the corresponding block pool at the datanode is deleted automatically. In the process of cluster up-gradation, each namespace volume is upgraded as a unit.

**Configuration files that are to be edited for sure while installing a hadoop cluster.**

Core-site.xml

HDFS-site.xml

YARN-site.xml

Xml