Dang Khoa Le

613760

ASSIGNMENT 2

1. **For NameNode, why it’s not necessary to store block locations persistently?**

Because block locations get changed very frequently and can be fully constructed by BlockReports which are sent by DataNodes even in the case of recovering from failure. By this way of getting information from DataNodes regularly, block locations are always up to date.

1. **Why is it important to make the NameNode resilient to failures?**

Because NameNode maintains the entire FileSystem metadata in HDFS and in the data flow of the system itself, read/write requests from client to HDFS must go to NameNode first to acknowledge a particular DataNode for read/write operations. A failure of NameNode in Hadoop 1.x makes HDFS unusable.

1. **What details are there in the FsImage file?**

FsImage file is a complete persistent checkpoint of filesystem metadata excluding block locations (Block to DataNode mapping).

1. **What is the purpose of the Secondary NameNode?**

Secondary NameNode merges the FsImage and the edits log file periodically (default is every hour) and keeps edits log size within a limit. It can also be considered as a backup of NameNode’s metadata.

1. **Does the NameNode stay in the safe mode until all under-replicated files are fully replicated? Why or why not?**

No, NameNode doesn’t needs to wait until all under-replicated files are fully replicated to exit safe mode. Because only a configurable percentage of safely replicated blocks is qualified, NameNode can exit the safe mode. Moreover, in the safe mode, replication of data blocks do not occur.

1. **What are the core changes in Hadoop 2.x compared to Hadoop 1.x? In other words, state the major differences between Hadoop 1 and Hadoop 2.**

Hadoop 2 provides punch of solutions to overcome the major drawbacks of Hadoop 1:

* Remove SPOF on NameNode, expand from one namespace per cluster to multiple namespaces per cluster, and even support namespace isolation: **High Availability** and **HDFS Federation**
* Remove SPOF on JobTracker, and added support for non-mapreduce type of processing (multitenancy) making MapReduce as a user library, or one of the applications residing in Hadoop: **YARN**
* Use of variable-sized Containers instead of fixed-size Slots mechanism: **High Cluster Utilization**
* **MRv2** (rebuilt from MRv1 to run on top of YARN)
* **Beyond Java**

1. **What is the difference between MR1 in Hadoop 1.0 and MR2 in Hadoop2.0?**

MR2 decouples MapReduce's *resource management* and *scheduling capabilities* from the *data processing component*, enabling Hadoop to support more varied processing approaches and a broader array of applications.

With MR2 (YARN) design, MapReduce is no longer at the core of Hadoop and is now a YARN application running in user space. This means that you can now run different versions of MR on the same cluster at the same time. Even more, Hadoop2 can now run interactive querying and streaming data applications simultaneously with MapReduce batch jobs.

1. **What is HDFS Federation? What advantage does it provide?**

HDFS Federation allows HDFS metadata to be shared across multiple NameNodes. It partitions the filesystem namespace over multiple separated NameNodes each of which manages a portion of the filesystem namespace. It also means HDFS Federation provides data isolation.

This comes with 2 main advantages:

* Aides HDFS scalability – there’s no longer memory limitation in NameNode bounding capacity of HDFS cluster.
* Data isolation and multiple namespace support - allowing different applications or teams to run their own NameNodes without fear of impacting other NameNodes on the same cluster.

1. **What is NameNode High Availability and how is it achieved in Hadoop 2?**

* The High Availability (HA) feature in Hadoop 2 addresses the NameNode SPOF problem by providing the option of running two redundant NameNodes in the same cluster in an Active/Passive configuration with a hot standby.
* This allows a fast failover to a new NameNode in the case that a machine crashes, or a graceful administrator-initiated failover for the purpose of planned maintenance.

The mechanism to achieve HA in Hadoop2:

* In a typical HA cluster, two separate machines are configured as NameNodes. At any point in time, exactly one of the NameNodes is in an Active state, and the other is in a Standby state.
* The **Active NameNode** is responsible for all client operations in the cluster, while the **Standby** is simply acting as a slave, maintaining enough state to provide a fast failover if necessary.
* In order for the Standby node to keep its state synchronized with the Active node, it is required that the two nodes both have access to a **shared directory on a shared storage device.**
* When any namespace modification is performed by the Active node, it durably logs a record of the modification to an edits log file stored in the shared directory.
* The Standby node is constantly watching this directory for edits, and as it sees the edits, it applies them to its own namespace.
* In the event of a failover, the Standby will ensure that it has read all the edits from the shared storage before promoting itself to the Active state.
* This ensures that the namespace state is fully synchronized before a failover occurs.
* In order to provide a fast failover, it is also necessary that the Standby node has up-to-date information regarding the location of blocks in the cluster.
* In order to achieve this, the DataNodes are configured with the location of both NameNodes and send block location information and heartbeats to both.

1. **What is the role of Application Master in YARN application execution?**

* It’s a short-life daemon process and one per application
* It’s a framework specific library and is tasked with negotiating resources from the ResourceManager and working with the NodeManager(s) to execute and monitor the tasks.
* It is created by the ResourceManager and is responsible for requesting containers to perform application-specific work.