Apache HBase

Brief by Surabhi Sarnot, Himanshu Agrawal

Why should you care?

Apache HBase is a distributed, master-slave, wide column (variable schema), key-value-pair database store that is linearly scalable. It works on top of Hadoop and does not require a well defined column structure. It allows real time and random read/write access to Big Data. It provides auto sharding (keeps data that is accessd together on a single cluster grouped by column families) and auto failure recovery mechanism that makes it highly scalable and reliable. It supports multiple versions of the same record, ie, supports taking snapshots of metadata to get previous or correct state form of data.

When to consider HBase?

- Application has a variable schema; each record may have slightly different columns.
- Large number of columns are null in each record.
- Key based access required to retrieve or store data.

Approach –

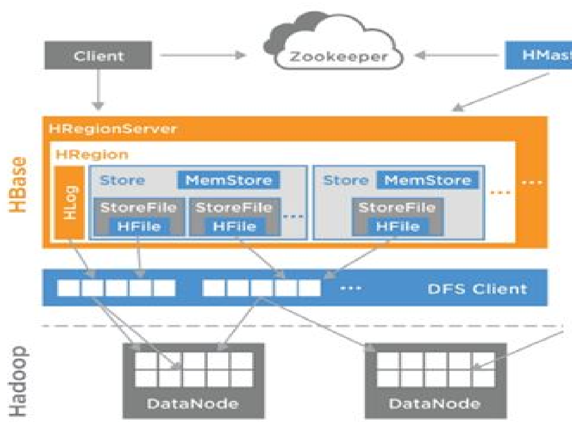

Figure 1: Apache HBase Architecture (Source)

HBase has three major components, **HMaster** Server responsible for coordinating Regions in the cluster and executing administrative operations; HBase **Region Server**, responsible to handle a subset of the table's data, **Zookeeper** acts as a centralized service for providing distributed synchronization and group services and maintaining configuration and naming information. **HFile** is used to store data in Hbase. The **Write Ahead Log** (WAL) is used to ensure that data changes are persisted to the database incase of RegionServer failure. It writes all HBase data changes to a file based storage which is used when Regionserver crashes before MemStore is flushed.

Results -

HBase has many similarities with Casandra, a NOSQL database also used for managing large datasets. Though they might seem similar at first, we take a look at some key differences that help us decide which one to use.

	Cassandra	**HBase**
Architecture	Masterless: does not have single point of failure	Master-based: single point of failure
Performance	Works best on static data	Handles “dynamic” data better
Security	User roles and conditions define accessibility	Visibility labels on datasets and then defines users/groups that can access them
Query Language	Cassandra Query Language(CQL)	Does Not have, need to involve extra technologies like hive etc
Data Consistency	Has issues	Strongly consistent

Table 1: Comparison between the main differences between choosing Cassandra and Apache HBase.

Pros/Cons-

Pros:

- Provides fast data reading and processing due to parallel access of distributed data.
- In case of HRegionServer failure it will do region assignment again thus providing auto failover.
- Hbase supports multiple versions of the same record. Supports taking snapshots of metadata to get previous or correct state form of data.

Cons:

- When only on HMaster is used, it will a single point of failure
- RDMS allows to create indexes on any column of the table, but HBase allows only to index key

Conclusion -

Apache HBase is an open source project that provides quick random access to Big Data with high reliability and scalability.