**IOT Device Streaming Data Pipelines**

**(Kafka + Hadoop (Cloudera) + Spark Structured Streaming + HBase + Impala)**

**Technologies:**

* Kafka
* Hive
* Impala
* Hadoop (Cloudera)
* Java EE
* Scala
* Spark Structured streaming
* Rest-API
* Nodejs
* HBase

**Introduction:**

Apache Kafka is a distributed publish-subscribe messaging system and a robust queue that can handle a high volume of data and enables you to pass messages from one end-point to another. Kafka is suitable for both offline and online message consumption. Kafka messages are persisted on the disk and replicated within the cluster to prevent data loss. Kafka is built on top of the Zoo-Keeper synchronization service. It integrates very well with Apache Storm and Spark for real-time streaming data analysis.

Benefits

Following are a few benefits of Kafka −

• Reliability − Kafka is distributed, partitioned, replicated and fault tolerance.

• Scalability − Kafka messaging system scales easily without down time.

• Durability − Kafka uses Distributed commit log which means messages persists on disk as fast as possible, hence it is durable.

• Performance − Kafka has high throughput for both publishing and subscribing messages. It maintains stable performance even many TB of messages are stored.

Kafka is very fast and guarantees zero downtime and zero data loss.

Spark Structured Streaming is a scalable and fault-tolerant stream processing engine built on the Spark SQL engine. You can express your streaming computation the same way you would express a batch computation on static data. The Spark SQL engine will take care of running it incrementally and continuously and updating the final result as streaming data continues to arrive

**Working:**

There are multiple IOT Simulators that send data every second to Kafka. Each IOT Simulator takes City Name and then crawls and parse google search to find its coordinates and temperature and update temperature every 30 min. I have created a Rest API end point which listens to the events fired by IOT Devices any time and pushes events to Apache Kafka which is installed on Cloudera Hadoop Cluster and uses the cloudera Cluster Zookeeper. Then Implemented the Spark Structured Streaming Job which reads data from Kafka Topic and performs operations on the data in real time. Then that once its transformed into entities is being pushed to HBase (No SQL) database. Finally created Impala Table on top of HBase Table and query it in SQL like syntax.

**Pre-Requisites:**

* Cloudera Quickstart VM
* Cloudera CDH 5+ installed
* Kafka must be Installed on Cloudera Quickstart VM
* Spark2 must be installed on Cloudera Quickstart VM
* Java 1.8 + must be installed and configured on each node of your cluster

Installation Steps:

1. Start Quickstart VM check all the services running properly. Obtain IP Address of your VM.
2. Replace the 'KAFKA\_HOST' value with IP Address in Kafka\_EndPoint\_API->server.js & Kafka\_EndPoint\_API->Create\_Topics.js
3. Run the 'Run\_Kafka\_End\_Point.bat' file. which will create the topic and will start the Rest API End point for Kafka.
4. 4. Run 3 instances of IOT\_Simulator which will ask for location and you can provide any city name it will pick up the City Coordinates and Temprature on its on by crawling and parsing google search.
5. create HBase table using HBase shell (type HBase Shell in cmd) to access Hbase shell.

create 'IOT\_Data\_Points','deviceId','temperature','location','time','jsonstring'

1. Now create hive table using hive shell (Type Hive in cmd) to access Hive shell and create Hive table using following command.

CREATE EXTERNAL TABLE hbase\_iot\_data\_points (rowID STRING, deviceId STRING, latitude INT,longitude INT, temperature INT, time TIMESTAMP,jsonstring STRING)

STORED BY 'org.apache.hadoop.hive.hbase.HBaseStorageHandler'

WITH SERDEPROPERTIES ("hbase.columns.mapping" = ":key,deviceId:deviceId,location:latitude,location:longitude,temperature:temperature,time:time,jsonstring:jsonstring")

TBLPROPERTIES("hbase.table.name" = "IOT\_Data\_Points");

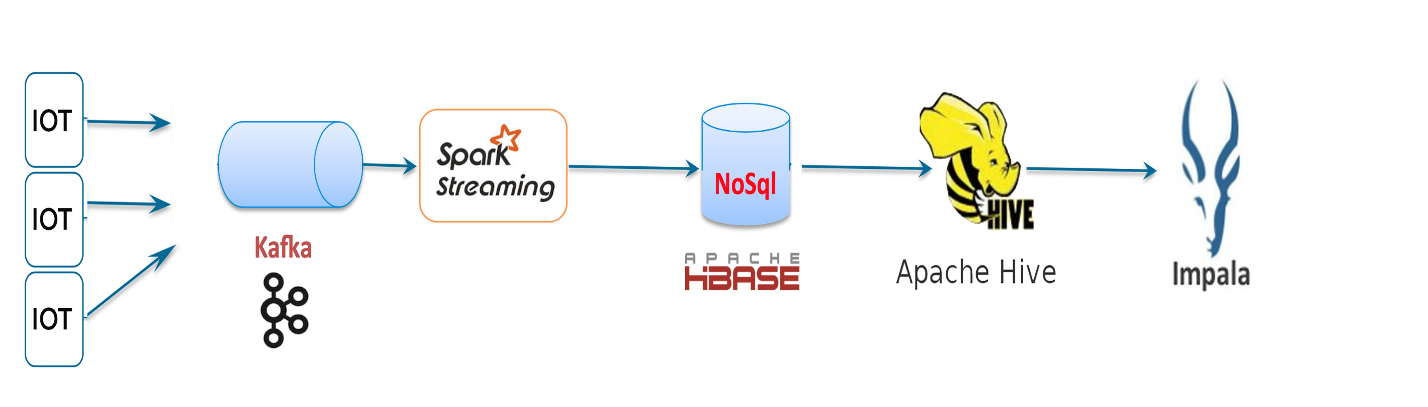
1. Now update impala metadata so that you can query HBase table from impala using following command

invalidate metadata hbase\_iot\_data\_points;

1. Create a jar file form the project 'SparkStructuredStreamingWithKafka' and copy it to some locatio on your VM using WinSCP.
2. Run Spark2-Submit job using following command by providing necessary dependencies and packages.

spark2-submit --packages org.apache.spark:spark-sql-kafka-0-10\_2.11:2.4.4,org.apache.spark:spark-streaming-kafka-0-10\_2.11:2.4.4,org.apache.kafka:kafka-clients:2.4.0,org.apache.hbase :hbase-client:1.2.0,org.apache.hbase:hbase-common:1.2.0,org.apache.hbase:hbase-protocol:1.2.0 --class SparkApp --master local /path/to/jarfile.jar

1. Finally Run the Impala-Shell to query the HBase table.
2. Alternatively, you can run the 'Run\_FrontEnd.bat' and the go to 'http://localhost:9898' to use the front-end web Interface for HBase and query data.

**Architecture:**

Demo Video

Please go to watch a live working Demo <https://www.youtube.com/watch?v=VZ84HRKnq4I> on YouTube (Must)