**Introduction to setup the IOT Pipeline via Eclipse Mosquitto/Kafka-connect to Mongo Datasink**

This part covers where we will see how to configure an IOT pipeline to receive IOT device data via mqtt publisher to mqtt topic configured for the respective device

1. We'll use [Docker Compose](https://docs.docker.com/compose/) to set up the infrastructure. That includes an MQTT broker as the source, Zookeeper, one Kafka broker as well Kafka Connect as middleware, and finally a MongoDB instance including a GUI tool as the sink.

You will need docker to be installed to run the entire infrastructure.

Attached is a docker-compose.yaml file which has to be executed by going to the path where it exists and running the command “docker-compose up”

1. Once the infrastructure is up , copy the connector files for mqtt and mongodb required by Kafka Connect. You should download them from
   1. <https://www.confluent.io/hub/confluentinc/kafka-connect-mqtt> and <https://www.confluent.io/hub/hpgrahsl/kafka-connect-mongodb>
2. After downloading the zip files ,extract them to a folder and then copy them to the kafka-connect container by running the below commands
   1. docker cp confluentinc-kafka-connect-mqtt-1.2.4 kafka-connect:/etc/kafka-connect/jars
   2. docker cp hpgrahsl-kafka-connect-mongodb-1.4.0 kafka-connect:/etc/kafka-connect/jars
3. Once the connector jars are loaded by kafka connect , we will proceed to configure the connectors via REST endpoint for the respective mqtt client and the mongodb sink for the flow from the iot device to the datasink to work e.g are the two commands
   1. *curl -d @connect-mqtt-arrayofswitches-source.json -H "Content-Type: application/json" -X POST* [*http://localhost:8083/connectors*](http://localhost:8083/connectors)

the argument in the curl command to the json file contains the below data. So you will have to create a json file with the name “*connect-mqtt-arrayofswitches-source.json”* and dump the below data

{

"name": "mqtt-arrayofswitches-source",

"config": {

"connector.class": "io.confluent.connect.mqtt.MqttSourceConnector",

"tasks.max": 1,

"mqtt.server.uri": "tcp://mosquitto:1883",

"mqtt.topics": "arrayofswitches",

"kafka.topic": "connect-arrayofswitches",

"value.converter": "org.apache.kafka.connect.converters.ByteArrayConverter",

"confluent.topic.bootstrap.servers": "kafka:9092",

"confluent.topic.replication.factor": 1

}

}

The values highlighted are unique for every connection

Name : name of the mqtt connection

Mqtt.topics: Is the topic to which the mqtt client will dispatch the low powered data

Kafka.topic: the Kafka topic which is connected to the datasink and the data is dumped

* 1. *curl -d @connect-mongodb-sensor-arrayofswitches-sink.json -H "Content-Type: application/json" -X POST http://localhost:8083/connectors*

the argument in the curl command to the json file contains the below data. So you will have to create a json file with the name “*connect-mongodb-sensor-arrayofswitches-sink.json”* and dump the below data

{

"name": "mongodb-iotsinkarrayofswitches",

"config": {

"connector.class": "at.grahsl.kafka.connect.mongodb.MongoDbSinkConnector",

"tasks.max": 1,

"topics": "connect-arrayofswitches",

"mongodb.connection.uri": "mongodb://mongo-db/sensor?retryWrites=true",

"mongodb.collection": "SensorData",

"key.converter": "org.apache.kafka.connect.json.JsonConverter",

"key.converter.schemas.enable": false,

"value.converter": "org.apache.kafka.connect.json.JsonConverter",

"value.converter.schemas.enable": false

}

}

The values highlighted are unique for every connection

Name : name of the mongodbsink connection

topics: Is the topic to which the mqtt data sink service will read the data from and dump to mongodb

mongodb.collection: the mongodb collection to which the sensor data is dumped for any IOT device. This same collection will be referenced by the REST service for pulling the different data as per the sensor id and other values for the respective service calls.

1. Now that the connectors are configured and setup to kafka-connect , we will run the batch job for the respective IOT device which dispatches the IOT sensor data every second to the mqtt topic and thus to the mongodb sink. I have created 4 sample sensor batch files and for the above connections the batch file is ***arrayofswitches.bat.*** The code of the batch file contains 2 important commands / parameters which are executed
   1. The Timestamp when the message is sent
   2. The docker command which is as below
      1. docker run -it --rm --name mqtt-publisharrayofswitches --network iot\_default efrecon/mqtt-client pub -h mosquitto -t "arrayofswitches" -m "{\"sensorId\":1002,\"message\":\"This is a arrayofswitches test - continuously\",\"value\":2,\"timestamp\":\""%dt%:%mytime%"\"}"

The highlighted parts are

* The name of the container to publish the data from the client
* the network used by docker i.e iot\_default , was at my end( yours can be obtained by running **docker network ls** command.)
* The name of the mqtt topic as configured in the mqtt connector
* The data sent by the sensor continuously . The fields are the ones that can be made dynamic in the batch file i.e the value and the timestamp is already dynamic.

Attached is the sample batch file

Once the batch executes it will show the ouput every second or as and how the services respond to the entire end to end flow. You can verify the mongo data by logging in to the dockerized mongo container and for the **SensorData** collection with the **Sensor** database instance.

**Introduction to setup Spring Boot REST Application to access the Services(Max/Min/Avg/Median)**

1. Please unzip the IOT application and via the import statement i.e mavenized project, include it in your Eclipse STS or any other IDE
2. Before executing ensure that the mongo endpoint in application.properties is the same as your dockerized mongo ip.
3. Once executed and it starts up , you can access the below services either via postman or I have even configured swagger UI to access the endpoints.
   1. [http://localhost:8080/readSensorMaximumValue/<single sensor id or multiple seperated by comma>/<startDate>/<endDate>](http://localhost:8080/readSensorMaximumValue/%3csingle%20sensor%20id%20or%20multiple%20seperated%20by%20comma%3e/%3cstartDate%3e/%3cendDate%3e) - Maximum value for the sensor/s in a given time frame
   2. [http://localhost:8080/readSensorMinimumValue/<single sensor id or multiple seperated by comma>/<startDate>/<endDate>](http://localhost:8080/readSensorMaximumValue/1000,1001,1003,1004/08-17-2020:22:10:21.85/08-17-2020:23:55:21.85) - Minimum value for the sensor/s in a given time frame
   3. [http://localhost:8080/readSensorAverageValue/<single sensor id or multiple seperated by comma>/<startDate>/<endDate>](http://localhost:8080/readSensorMaximumValue/1000,1001,1003,1004/08-17-2020:22:10:21.85/08-17-2020:23:55:21.85) - Average value for the sensor/s in a given time frame
   4. http://localhost:8080/readSensorMedianValue/<single sensor id or multiple seperated by comma>/<startDate>/<endDate> - Median value for the sensor/s in a given time frame

You have to ensure that

* The sensor id is the one sent by you from the batch file in the previous section
* The date is in mm-dd-yyyy:hh:mi:ss format
* The data is sorted for he above calls based on the value sent for the sensor data i.e is long or numeric value as sent and dumped

**Drawbacks/ Limitations of the implementation**

* The IOT data has to be configured in a very rigid way and even the data/values/readings sent are now static. Will have to randomize via the script
* The IOT pipeline kafka services are managed by Zookeeper and at many times if idle , the services shutdown or take time to get back up. Had to restart the services many times
* The Swagger UI randomly works for the REST services i.e if you check <http://localhost:8080/swagger-ui.html#/> the services were visible but sometimes has to be regenerated

P.S Ive renamed the batch files with extension to txt due to security reason/gmail attachment. Please rename them back to bat and then run them ☺