**Spark Streaming + Kafka Assignment**

**Pavithrananda Prabhu S**

1. **Objective**
2. Write the Kafka Producer which produces output in the below format

message format: key: Key1 val: {"TIMESTAMP": "2017-02-25T04:44:18", "val": 0, "key": "Key1”}

frequency of message: 30 seconds

1. Write the spark streaming app to read the Data from Kafka Topic / Perform aggregation / Write back the data in to Kafka.

Aggregation message format: key: Key3 val: {"count": 4, "TIMESTAMP": "2017-02-25T04:42:00", "sum": 9, "ts": ["2017-02-25T04:42:48", "2017-02-25T04:42:20", "2017-02-25T04:43:24", "2017-02-25T04:43:48"], "key": "Key3", "vals": [4, 1, 1, 3], "mean": 2.25}

Kafka\_output\_Topic : TEST\_AGGREGATED

1. **High Level Design**

Create the necessary

Input topic and

output topic in kafka

( kafkaTopic.py)

KafkaProducer to Write

Data in to Kafka\_input\_topic

PRABHU\_INPUT\_TOPIC

( kafkaProducer.py )

Spark Streaming

-

Read the data from

Kafka\_input\_topic

( sparkStreaming.py )

Spark Streaming

-

Perform the groupbykey

operation and create

Aggregates for each key

( sparkStreaming.py )

SparkStreaming

-

Write the aggregated data back

To kafka\_output\_topic

TEST\_AGGREGATED

( sparkStreaming.py )

1. **Design breakdown**

1. **kafkaTopic.py**

It creates the necessary topics required for the kafkaProducer.py and sparkStreaming.py

It is run using the command python kafkaTopic.py create (create topic) /python kafkaTopic.py list (list topics) / python kafkaTopic.py delete (delete topics )

**Input\_topic\_name :- PRABHU\_INPUT\_TOPIC**

created using the below configuration.

kafka-topics --zookeeper 9.109.184.69:2181 --create –topic PRABHU\_INPUT\_TOPIC --partitions 5 --replication-factor 1

This input topic is created using 5 partitions the reason being to introduce parallelism in Spark Streaming

data with key1 goes in to partition 0 of PRABHU\_INPUT\_TOPIC

data with key2 goes in to partition 1 of PRABHU\_INPUT\_TOPIC

data with key3 goes in to partition 2 of PRABHU\_INPUT\_TOPIC

data with key4 goes in to partition 3 of PRABHU\_INPUT\_TOPIC

data with key5 goes in to partition 4 of PRABHU\_INPUT\_TOPIC

**Output\_topic\_name :- TEST\_AGGREGATED**

created using the below configuration.

kafka-topics --zookeeper 9.109.184.69:2181 --create --topic TEST\_AGGREGATED --partitions 1 --replication-factor 1

output topic is created with only one partition for this assignment but if needed we can also set this to 5

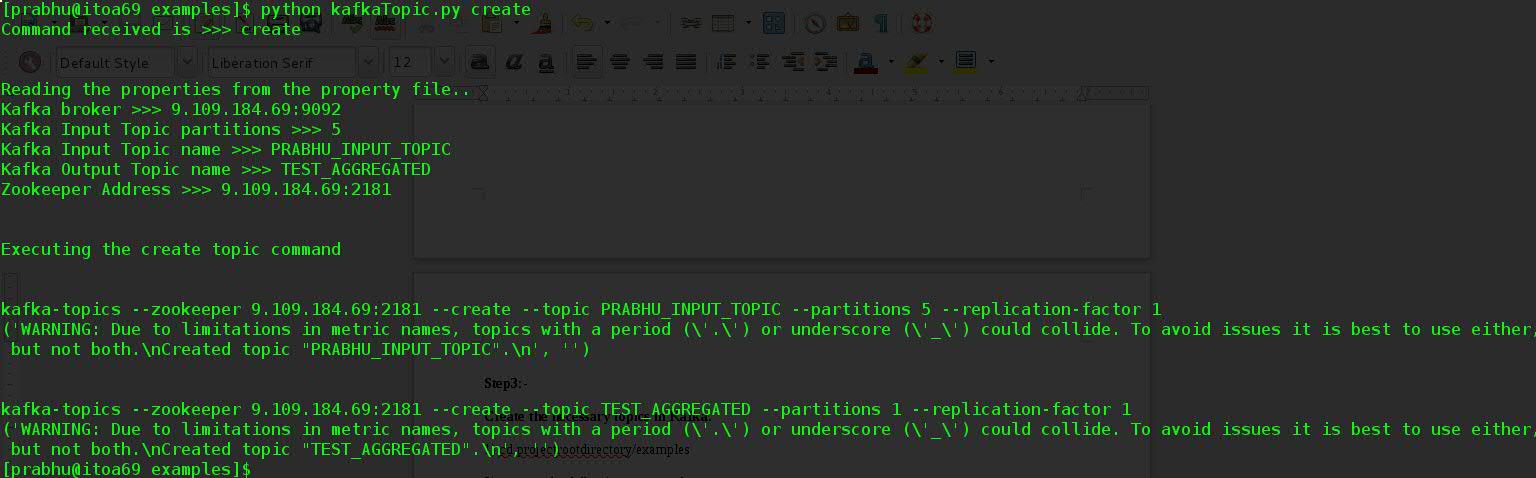
**Program Flow :-**

1) Read the configurations from the config/Configfile.properties file

2) Check which parameter is given as input create/list/delete based on which issue the necessary command

3) Use the subprocess package in python to issue the necessary commands to create / list / delete topics in Kafka.

**Sample Output :-**

****

1. **kafkaProducer.py**

This program produces the required data in Kafka\_input\_topic

**Messages written are in the below format**

key: Key1 val: {"TIMESTAMP": "2017-02-25T04:44:18", "val": 0, "key": "Key1”}

**Kafka topic to which messages are written :-** PRABHU\_INPUT\_TOPIC

**Input\_Topic Partitions :-** 5

data with key1 goes in to partition 0 of PRABHU\_INPUT\_TOPIC

data with key2 goes in to partition 1 of PRABHU\_INPUT\_TOPIC

data with key3 goes in to partition 2 of PRABHU\_INPUT\_TOPIC

data with key4 goes in to partition 3 of PRABHU\_INPUT\_TOPIC

data with key5 goes in to partition 4 of PRABHU\_INPUT\_TOPIC

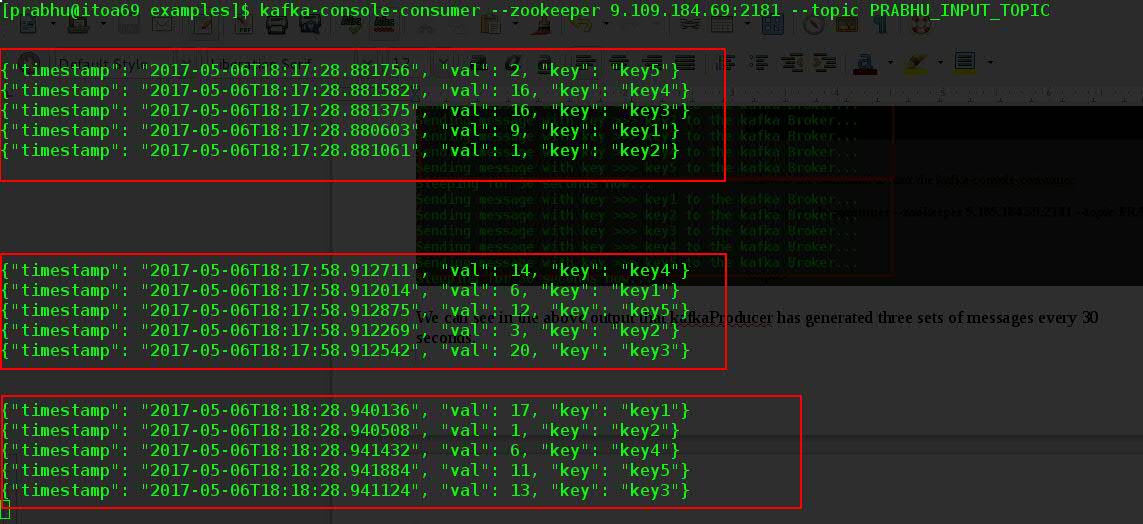
**Program Flow:-**

1) Program basically has a forever ending while loop

2) sleep(30) program sleeps every 30 seconds

3) After 30 seconds the program generates the message of the format specified above and writes the data to the kafka Topic PRABHU\_INPUT\_TOPIC

**Sample Output :-**



1. **sparkStreaming.py**

This is the Spark Streaming program which basically

1) reads data from the kafka\_input\_topic.

2) produces aggregates for 2 mins window.

3) writes data to kafka\_output\_topic.

**kafka\_input\_topic :-** PRABHU\_INPUT\_TOPIC

**kafka\_output\_topic :-** TEST\_AGGREGATED

Sample Command to Start the stream

**spark-submit --packages org.apache.spark:spark-streaming-kafka-0-8\_2.11:2.0.0 sparkStreaming.py**

**Program Flow :-**

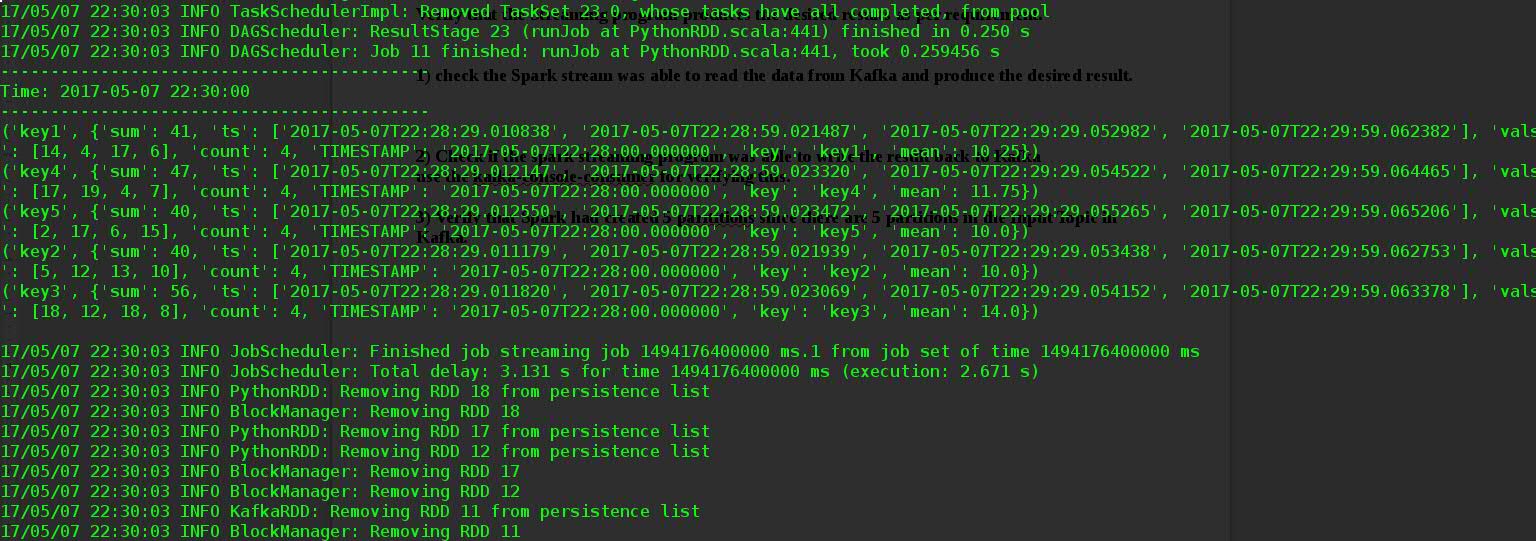
1) Read the data from the kafka\_input\_topic

2) Use the groupbyKey transformation function to group the data for each key

3) Use the function get\_desired\_assignment\_output to get the desired output as per the assignement for each key

4) Use foreachRDD to write the data back to kafka\_output\_topic.

**Sample output :-**



1. **Project Directory Structure**

Project basically has the following folders each having its purpose.

1) config directory => Contains the configurations necessary for the Project

2) examples directory => Contains the main programs

3) functionkit directory => Contains all the functions used by the programs in the examples directory

4) doc directory => contains the documents related to the Project.

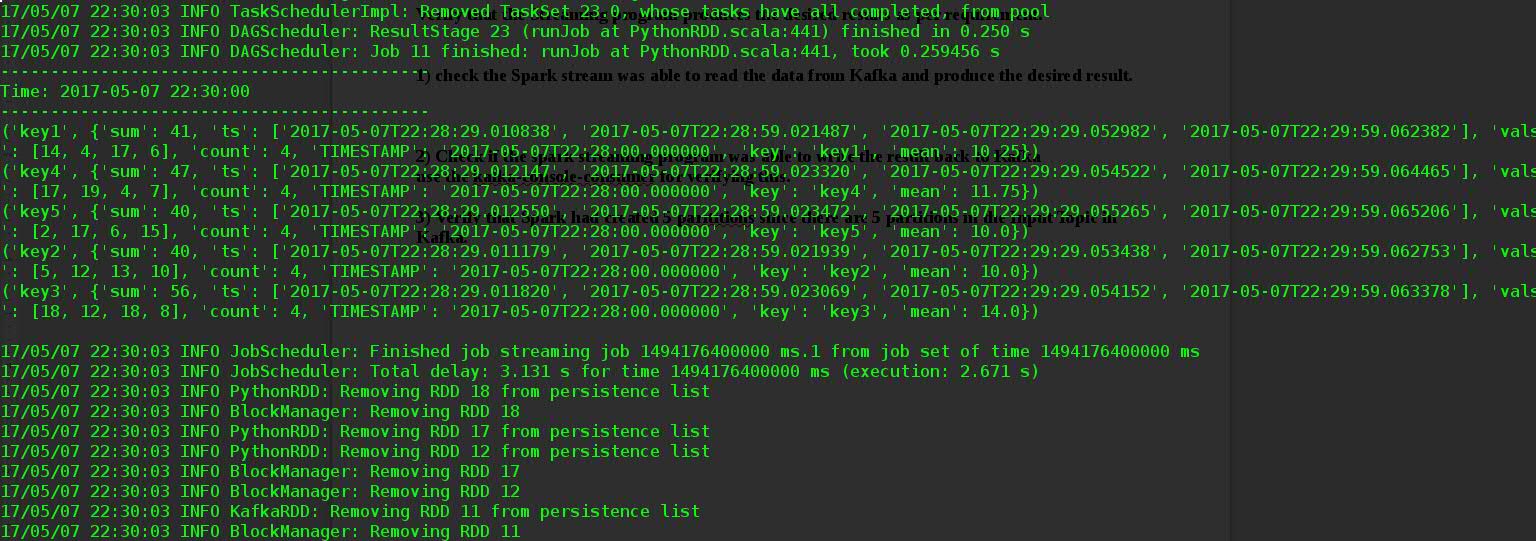
1. **Validation**

Detailed step by step on executing the program is documented in the following documents

1) Kafka Steps.docx

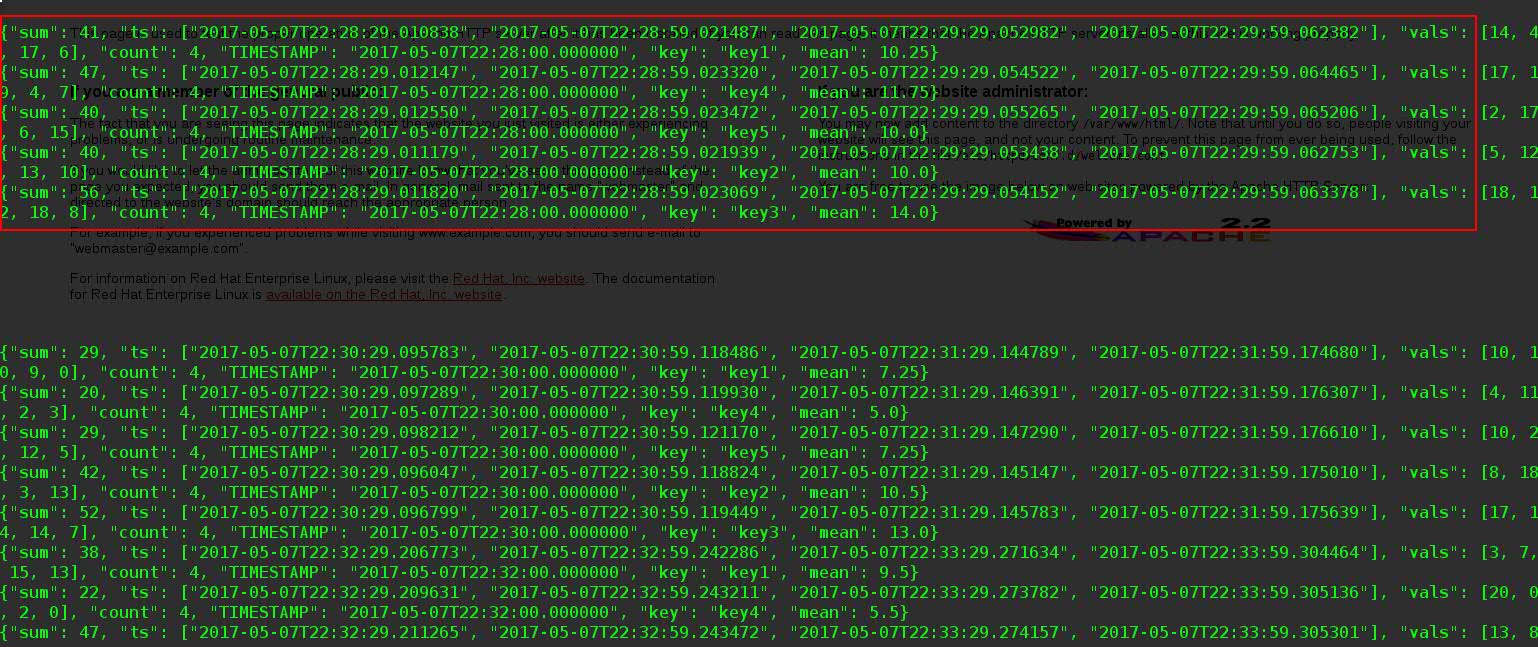
2) Spark Steps.docx

**Final Spark Output**



we can see from the above output that it had produced the output as per the requirement.

**Final Kafka\_output\_topic Output**



Above is the output from the Kafka-console-consumer and we can see that the output generated from the spark-streaming is available in the Kafka-output-topic which is TEST\_AGGREGATED**.**