Big Data Technologies

Assignment #9

Ashmita Gupta (A20512498)

Exercise 1) 5 points

Read the article “Real-time stream processing for Big Data” available on the blackboard in the ‘Articles’ section and then answer the following questions:

a) (1.25 points) What is the Kappa architecture and how does it differ from the lambda architecture?

Ans.

1) **Kappa**: As per the article on the kappa architecture, the precomputation of data does not happen on regular basis in the batch layer. All the computation is done in the stream processing system. The re-computation is performed only when the business logic changes by replaying past data. For this, the Kappa Architecture uses a powerful stream processor which can deal with data at a faster speed than the speed with what the data comes in.
2) **Lambda**: Lambda architecture is a system consisting of three layers: Batch, Speed, and Serving layer. It targets both the volume and velocity challenge of big data at the same time. It has both a batch-oriented system and a real-time system.

b) (1.25 points) What are the advantages and drawbacks of pure streaming versus micro-batch real-time processing systems?

Ans.

1) Storm and Samza are pure stream-oriented systems with very low latency and somewhat high per-item costs.
2) Batch-oriented systems offer unmatched resource efficiency at the tradeoff of unreasonably high latency for real-time applications. Data is buffered and processed in batches in micro-batch real-time processing systems. It improves efficiency while also increasing the time an individual item spends in the data flow. Storm Trident and Spark Streaming are two examples of this type.

c) (1.25 points) In few sentences describe the data processing pipeline in Storm.

Ans.

1) The topology in Storm refers to a data pipeline or application. Spouts are the nodes that take data and so start the data flow in the topology. Spouts output tuples to bolts, which execute processing, write data to external storage and may transmit tuples further downstream.
2) Data flow between nodes is controlled by storm groupings. Storm distributes spouts and bolts in a round-robin fashion by default, but the scheduler can be modified to accommodate cases in which a specific processing step must be performed on a specific node. Storm offers the option of at least one processing via an acknowledgment feature that maintains the processing status of every

single tuple as it travels through the topology. It does not ensure the sequence in which tuples are processed.

d) (1.25 points) How does Spark streaming shift the Spark batch processing approach to work on real-time data streams?

Ans.
It does so by chunking the stream of incoming data items into small batches, changing them into DDs, and processing them as usual, Spark streaming shifts the batch-processing method towards real-time requirements. Further, it automatically manages data flow and distribution.

Exercise 2) 5 points (extra credit; if you don't want to try or if you try and can't get things to work, this won't impact your score negatively)

Refer to the python-Kafka Documentation from the Free Books and Chapters section of our blackboard site

Step A – Start an EMR cluster

Starting an EMR Cluster

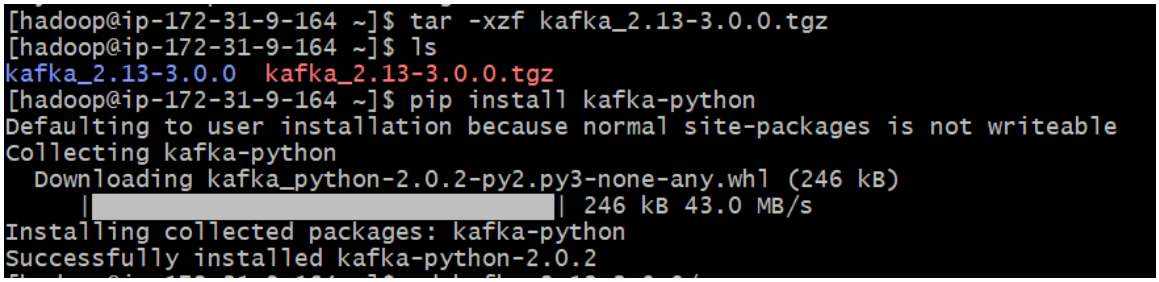
```
gashm@Ashmita MINGW64 ~/OneDrive/Desktop/Big Data
$ ssh -i emr-key-pair-2.pem hadoop@ec2-3-235-107-43.compute-1.amazonaws.com
Last login: Tue Nov  7 06:31:24 2023

       __|  __|_  )
       _|  (     /   Amazon Linux 2 AMI
      ___|\___|___|

https://aws.amazon.com/amazon-linux-2/
44 package(s) needed for security, out of 71 available
Run "sudo yum update" to apply all updates.

EEEEEEEEEEEEEEEEEEE MMMMMMMM           MMMMMMMM RRRRRRRRRRRRRRR
E::::::::::::::::::E M:::::::M         M:::::::M R::::::::::::::R
EE:::::EEEEEEEEE:::E M::::::::M       M::::::::M R:::::RRRRRR:::::R
  E::::E       EEEEE M:::::::::M     M:::::::::M RR::::R      R::::R
  E::::E             M::::::M:::M   M:::M::::::M   R:::R      R::::R
  E:::::EEEEEEEEEE   M:::::M M:::M M:::M M:::::M   R:::RRRRRR:::::R
  E::::::::::::::E   M:::::M  M:::M:::M  M:::::M   R:::::::::::RR
  E:::::EEEEEEEEEE   M:::::M   M:::::M   M:::::M   R:::RRRRRR::::R
  E::::E             M:::::M    M:::M    M:::::M   R:::R      R::::R
  E::::E       EEEEE M:::::M     MMM     M:::::M   R:::R      R::::R
EE:::::EEEEEEEE::::E M:::::M             M:::::M   R:::R      R::::R
E::::::::::::::::::E M:::::M             M:::::M RR::::R      R::::R
EEEEEEEEEEEEEEEEEEE MMMMMMM             MMMMMMM RRRRRRR      RRRRRR
```

Step B – Copy the Kafka software to the EMR master node

Moving kafka file to home/hadoop directory

Step C – Install the Kafka software and start it

Installing Kafka python package

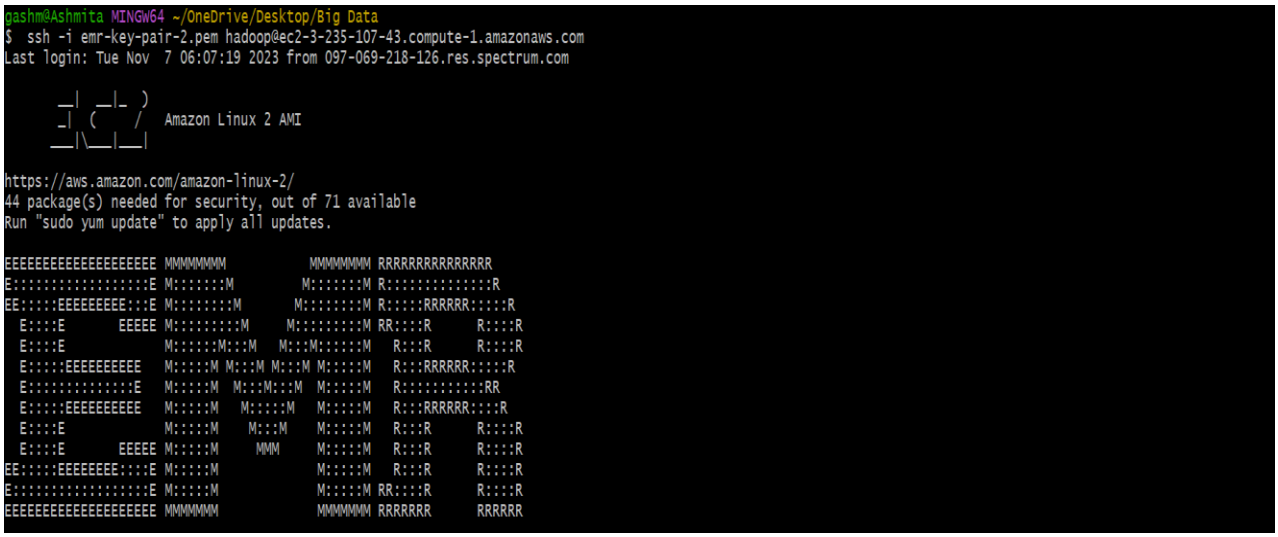


Running Command- bin/zookeeper-server-start.sh config/zookeeper.properties &

Running Command- bin/kafka-server-start.sh config/server.properties &

Step D – Prepare to run Kafka producers and consumers

2nd EMR Connection (Producer)



3rd EMR Connection (Consumer)

```
gashm@Ashmita MINGW64 ~/OneDrive/Desktop/Big Data
$  ssh -i emr-key-pair-2.pem hadoop@ec2-3-235-107-43.compute-1.amazonaws.com
Last login: Tue Nov  7 06:07:19 2023 from 097-069-218-126.res.spectrum.com

       __|  __|_  )
       _|  (     /   Amazon Linux 2 AMI
      ___|\___|___|

https://aws.amazon.com/amazon-linux-2/
44 package(s) needed for security, out of 71 available
Run "sudo yum update" to apply all updates.

EEEEEEEEEEEEEEEEEEEE MMMMMMMM           MMMMMMMM RRRRRRRRRRRRRRR
E::::::::::::::::::E M::::::M         M::::::M R::::::::::::::R
EE::::::EEEEEEEEE:::E M:::::::M       M:::::::M R::::::RRRRRR:::::R
  E::::E       EEEEE M::::::::M     M::::::::M RR::::R      R::::R
  E::::E             M::::::M:::M   M:::M::::::M   R:::R      R::::R
  E::::::EEEEEEEEEE  M:::::M M:::M M:::M M:::::M   R:::RRRRRR:::::R
  E::::::::::::::E   M:::::M  M:::M:::M  M:::::M   R::::::::::RR
  E::::::EEEEEEEEEE  M:::::M   M:::::M   M:::::M   R:::RRRRRR::::R
  E::::E             M:::::M    M:::M    M:::::M   R:::R      R::::R
  E::::E       EEEEE M:::::M     MMM     M:::::M   R:::R      R::::R
EE::::::EEEEEEEE:::E M:::::M             M:::::M   R:::R      R::::R
E::::::::::::::::::E M:::::M             M:::::M RR::::R      R::::R
EEEEEEEEEEEEEEEEEEEE MMMMMMM             MMMMMMM RRRRRRR      RRRRRR
```

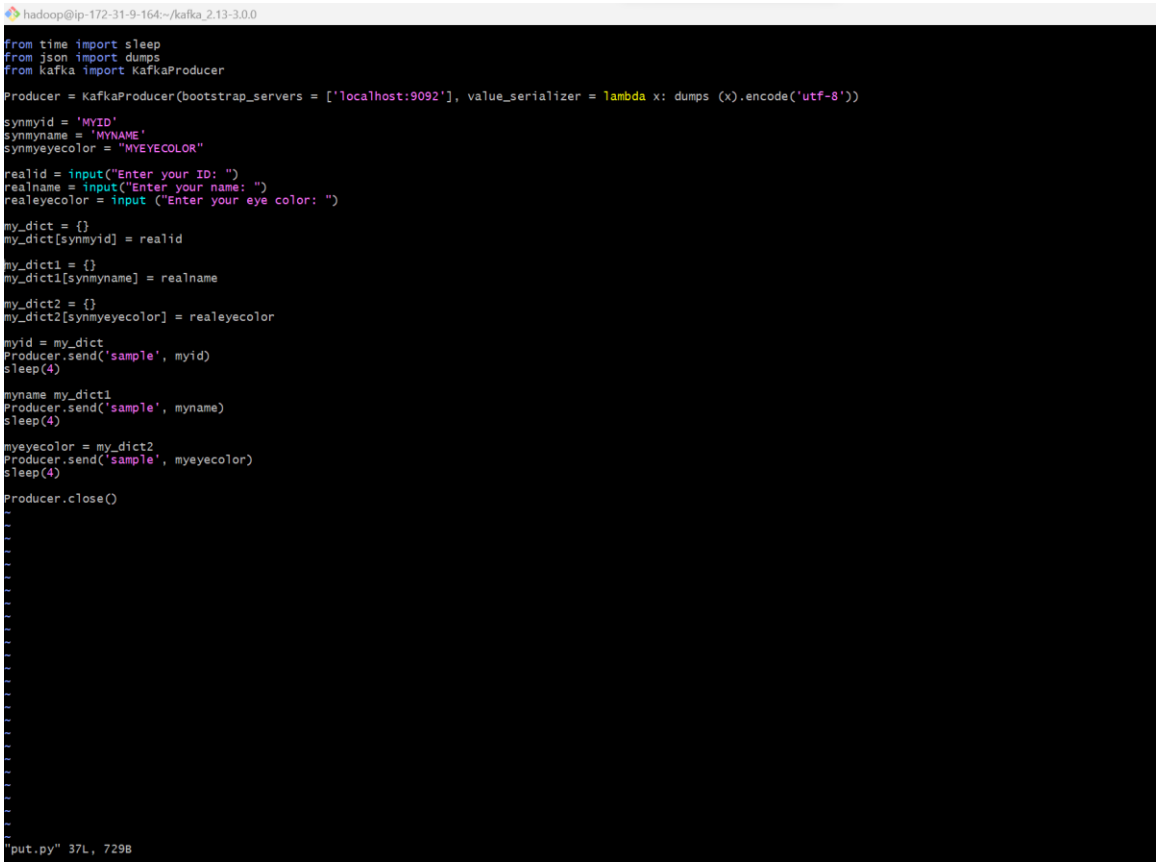
Step E – Creating a Kafka topics

```
[hadoop@ip-172-31-9-164 ~]$ cd kafka_2.13-3.0.0
[hadoop@ip-172-31-9-164 kafka_2.13-3.0.0]$ bin/kafka-topics.sh --create --replication-factor 1 --partitions 1 --bootstrap-server localhost:9092 --topic sample
Created topic sample.
[hadoop@ip-172-31-9-164 kafka_2.13-3.0.0]$ bin/kafka-topics.sh --create --replication-factor 1 --partitions 1 --bootstrap-server localhost:9092 --topic ashmita_gupta
WARNING: Due to limitations in metric names, topics with a period ('.') or underscore ('_') could collide. To avoid issues it is best to use either, but not both.
Created topic ashmita_gupta.
[hadoop@ip-172-31-9-164 kafka_2.13-3.0.0]$ bin/kafka-topics.sh --list --bootstrap-server localhost:9092
ashmita_gupta
sample
```

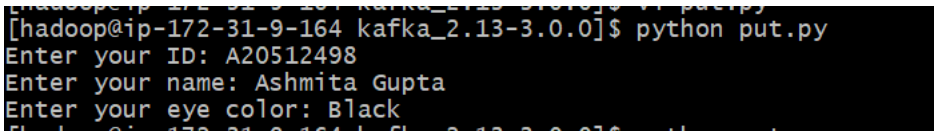
Producer Terminal

Key	Value
'MYID'	Your student id
'MYNAME'	Your name
'MYEYECOLOR'	Your eye color (make it up if you can't remember)

vi put.py



python put.py



Consumer Terminal
vi get.py

python get.py

```
[hadoop@ip-172-31-9-164 kafka_2.13-3.0.0]$ python get.py
key=MYID value=A20512498
key=MYNAME value=Ashmita Gupta
key=MYEYECOLOR value=Black
```