Thank you for inviting me to this challenge . On the first glance every things look ambiguous, and so I decided to give my best , since giveup is not an option and it will either be a learning or a success. That kept me going till the very end ….

With that Said, Let's Get Started:

Installing Kafka

This was the hardest part because I could not find the right way to execute kafka on my machine .

Approach 1 : Cygwin

As a first approach , using windows system , I tried to do it via Cygwin but had numerous problems . Through cygwin I was having the errors with environment including java virtual machine , and configurations in kafka-run-class.sh . There I changed several paths and also tried delete obsolete stuff. I was basically stuck with Cygwin setup with Kafka that didn't work.

Approach 2 :

The next approach was to Install Linux Bash Shell on Windows 10 . So I can run kafka commands from documentation in Unbuntu . I had few problems here too and I follwoied all the instructions in the following links :

https://itsfoss.com/install-bash-on-windows/

But unfortunately there were some assembly references missing and I figured out that in order to enable **Windows Subsystem for Linux** on my machine , I need to reinstall the windows 10. That's was not an option for me for several reasons. So proceeded with approach 3 .

Approach 3 :

This approach was running on windows cmd using the batch files .

The latest kafka version which was giving lots of problems to start zookeepr and kafka . After lots of research and reading stack overflows I installed

2.11-0.9.0.1 version of kafka. After installing zookeeper separately I was able to run zookeeper and kafka *2.11-0.9.0.1*.

ZooKeeper Installation

1. Go to your ZooKeeper config directory. For me its *C:\zookeeper\apache-zookeeper-3.7.0-bin\conf*

2. Rename file “zoo_sample.cfg” to “*zoo.cfg”*

3. Open zoo.cfg in any text editor

4. Find and edit dataDir=/tmp/zookeeper to C:/zookeeper/apache-zookeeper-3.7.0-bin/data

5. Add an entry in the System Environment Variables as we did for Java.

- Add ZOOKEEPER_HOME = C:/zookeeper/apache-zookeeper-3.7.0-bin/data to the System Variables.

- Edit the System Variable named “Path” and add ;%ZOOKEEPER_HOME%\bin;

6. You can change the default Zookeeper port in zoo.cfg file (Default port 2181).

7. Run ZooKeeper by opening a new cmd and type zkserver .

This will start the zookeeper .

Setting Up Kafka

1. Go to your Kafka config directory. For me its *C:\kafka\kafka_2.11-0.9.0.1\config*

2. Edit the file “server.properties.”

3. Find and edit the line log.dirs=/tmp/kafka-logs” to “log.dirs=C:/kafka/kafka_2.11-0.9.0.1/kafka-logs.

Running a Kafka Server

ensure that your ZooKeeper instance is up and running before starting a Kafka server.

1. Go to your Kafka installation directory: *C:\kafka\kafka_2.11-0.9.0.1*

2. Open a command prompt here

3. Now type .\bin\windows\kafka-server-start.bat .\config\server.properties and press Enter.

Now your Kafka Server is up and running, you can create topics to store messages.

Creating Topics

1. create a topic with the name *"my_topic"* and a replication factor of 1, as we have only one Kafka server running.

2. Open a new command prompt in the location *C:\kafka\kafka_2.11-0.9.0.1\bin\windows.*

3. Type the following command and hit Enter:

kafka-topics.bat --create --zookeeper localhost:2181 --replication-factor 1 --partitions 1 --topic my_topic

Creating a Producer and Consumer to Test Server

1. Open a new command prompt in the location *C:\kafka_2.11-0.9.0.0\bin\windows*

2. To start a producer type the following command:

kafka-console-producer.bat --broker-list localhost:9092 --topic my_topic

3. Again open a new command prompt in the same location as *C:\ kafka\kafka_2.11-0.9.0.1\bin\windows*

4. Now start a consumer by typing the following command:

kafka-console-consumer.bat --zookeeper localhost:2181 --topic my_topic

Now the Kafka setup is done. We add the test data to topic .

Use the Kafka producer from kafka itself to send test data to topic :

You can stream it using pipe operator in linux with the following command :

kafka-console-producer.sh --broker-list localhost:9092 --topic my_topic--new-producer < stream.jsonl

However in windows I didn't find the equivalent and used a my own generated different json file

Create a small app that reads this data from kafka and prints it to stdout

- find the code in two python solution files for producer.py and consumer.py in base dir.

Find a suitable data structure for counting and implement a simple counting mechanism, output the results to stdout

- Find the solution in file named Counting.py in base dir

How can you scale it to improve throughput?

The main way we scale data consumption from a Kafka topic is by adding more consumers to the consumer group. It is a common operation for Kafka consumers to do high-latency operations such as writing to databases or a time-consuming computation.

Selection of data structure :

Dictionary **Data structure** is used because the insert and search from dictionary both have time complexity of O(1) and it will be most performant in comparison to other data structures.

Coping with crashes :

- If you are running your own hardware infrastructure, make sure that you build in redundancy for networking equipment, storage, racks, and power supplies.
- List the failures that the system should tolerate. Tool to address these issues could be Multi-Region Clusters with Automatic Observer Promotion and Cluster Linking
- Distribute your brokers so that they can survive the failure of any one piece of infrastructure.
- Monitor your hardware and software infrastructure, and ensure that you have alerting set up. This way, you can detect and remedy problems quickly.
- Configure your topics with the appropriate replication settings
- Upgrade your clusters and client libraries at a regular cadence.
- Ensure that your deployments are automated and configuration is kept under version control. Nobody should be able to update broker configurations directly.

Performance and throughput

1. ***Compression rate***

A higher compression rate indicates greater efficiency

2. ***Request latency average***

The average request latency is a measure of the amount of time between when KafkaProducer.send() was called until the producer receives a response from the broker.

The producer linger.ms value determines the maximum amount of time it will wait before sending a message batch

The default value of linger.ms is 0 ms; setting this to a higher value can increase latency, but can also help improve throughput

Monitor request latency to ensure it doesn't rise beyond an acceptable limit.

Modifying batch.size in your producer configuration can lead to significant gains in throughput. Increase batch size.

3. ***I/O wait time : should be minimum***

4. ***records consumed rate :*** **this should be constant**

5. ***fetch rate***

In a healthy consumer, the minimum fetch rate will usually be non-zero, so if you see this value dropping, it could be a sign of consumer failure

6. **Zookeeper**

monitoring ZooKeeper is key to maintaining a healthy Kafka cluster. you should deploy ZooKeeper in a high-availability configuration called an ensemble

Error mecahnisms :

setting errors.tolerance = all will enable Kafka Connect to just ignore bad messages.

The most simplistic approach to determining if messages are being dropped is to tally the number of messages on the source topic with those written to the output:

kafkacat -b localhost:9092 -t my_topic -o beginning -C -e -q -X enable.partition.eof=true | wc –l

Choice of serialization

Json in general is easy to implement, supported natively by some DBs like MongoDB, PostgreSQL, and others. Any technology stack is able to parse or produce it. JSON format is heavily used and has many advantages.

However Plain text formats, especially JSON, have some problems with number precision.

Binary formats :

- The binary format is, more compressed, so storage usage will be lower.
- it could save on disk space
- the prices of fast and large SSD drives for DB purposes the actual costs are significant

We can use AVRO. Avro is an open source project that provides data serialization and data exchange services for Apache Hadoop.

Avro, is much easier to use.

The cost of this is that you will need to provide both reader and writer schema to deserialize anything.

Advantages of Avro

1. It has a direct mapping to and from JSON
2. It has a very compact format.
3. It is very fast.
4. It has great bindings for a wide variety of programming languages so you can generate Java objects that make working with event data easier, but it

does not require code generation so tools can be written generically for any data stream.
5. It has a rich, extensible schema language defined in pure JSON
6. It has the best notion of compatibility for evolving your data over time.

Avro is best for simple domains with mostly primitive types.

Remarks:

Over all the assignment was challenging, and the fun part was enjoying the complete journey and not just focusing on the end goals. I believe that in order to achieve success the laser focus on small tiny steps or individual units of big process would be a more wiser approach than trying to accomplish the end goal as whole.

References :

https://kafka.apache.org/documentation/

https://linuxhint.com/read_data_kafka_python/

https://stackoverflow.com/questions/64247670/how-to-count-number-of-records-message-in-the-topic-using-kafka-python

https://stackoverflow.com/questions/43282898/cant-consume-json-messages-from-kafka-using-kafka-pythons-deserializer

https://stackoverflow.com/questions/58841643/printing-top-few-lines-of-a-large-json-file-in-python

https://stackoverflow.com/questions/54672599/json-file-data-into-kafka-topic

https://itsfoss.com/install-bash-on-windows/

https://stackoverflow.com/questions/33273587/how-to-write-a-file-to-kafka-producer

https://stackoverflow.com/questions/25037263/apache-kafka-error-on-windows-couldnot-find-or-load-main-class-quorumpeermain

https://stackoverflow.com/questions/28484398/starting-zookeeper-cluster-error-could-not-find-or-load-main-class-org-apache/56631033#56631033

https://stackoverflow.com/questions/47168342/kafka-1-0-stops-with-fatal-shutdown-error-logs-directory-failed

https://stackoverflow.com/questions/67332490/after-zookeeper-audit-is-disable-stopped-working

https://stackoverflow.com/questions/68527801/zookeeper-server-expiring-session-timeout-exceeded-org-apache-zookeeper-server

https://stackoverflow.com/questions/37186197/errorcould-not-create-the-java-virtual-machine-errora-fatal-exception-has-occu/49894503

https://stackoverflow.com/questions/34081336/classpath-is-empty-please-build-the-project-first

https://stackoverflow.com/questions/51644409/kafka-broker-fails-because-all-log-dirs-have-failed