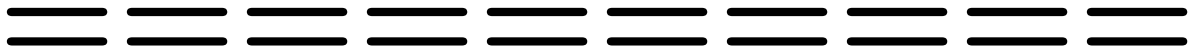
Apache

Kafka



=> It is java based messaging technique .. which can be implemented in different languges

=> It supports to use different technologies and concepts for messaging activity

like hadoop, spark, scala and etc..

(kafka integration is possible with multiple technologies of java and non-java env..)

=> Apache kafka is all about

Store + Process + Integrate (Transfer)

=>Kafka is basically used to transfer (integrate) data/messsages between multiple

complex Apps/systems using Cluster design.

cluster :: set of similar items is called cluster.

ac1

ac2

ac3

AC cluster

plot1

plot2

plot

Plot Cluster

=>Kafka intergration /interaction with non-java applications is possible with the support of REST calls.

=> Karfka is protocol Independent ... we can write code to send messages among the applications using http, tcp,ftp and etc.. protocols.

the

=> Kafka allows to take multiple message broker s/ws (MOM s/ws) at a time as cluster having support for Horizontal scaling giving the following advantages..

a) Fast data transfer for large data set /messagess

b) No data loss even one messsage broker s/w or MOM s/w is down

becoz other broker s/w or MOM s/w takes care of messages. the support of

c) It takes apache zookeeper to handle load balancing among the multiple instances of message broker s/w or MOM s/ws

apache zookeepr is like netflix eureka server..

Producer#1

Instance#1

of Message

broker server (MOM s/w

Instance#2

of Message

broker server IMOMS/W

Producer#2

Msg1

Msg2

Instance#3

of Message

Message broker

broker serve (MOMS/W)

server cluster

MOM s/w cluster

Producer#n

MOM s/w

Apache kafka

(can maintain cluster of Message brokers)

MOM s/w

MOM s/w

consumer#1

Msg3

consumer#n

consumer#2

Kafka Message Broker / Broker server / Mom Server / MOM software

========

It behaves like MOM/Broker to recieve messages/data from sender, to hold them as needed and to deliver to Consumer..

=>When kakfa s/w is started one instance of Message broker will be created automatically.. and it can be increased as needed.

of

=> collection message broker instances togather is called kafka message broker cluster .. and

we can add any no.of instances in one cluster .. i.e there is no limit to add instances.

=>Apache Zookeepr is resposible to manage message broker instances of cluster by applying Load Balance support and it is also called Bootstrap server becoz it is responsbile to create cluster having single instance and increasing the instances as needed.

(setup)

(Zookeeper)

=>Apache kafka Eco System = Boostrap server + message broker cluster.

=> The apache kafka Message broker gets Message from Sender /Publisher App in Serialized format and delivers to consumer/subscriber in DeSerialized format.

Kafka software (Kafka echo system)

(Zookeeper)

Bootstrap

server

Sender /Publisher App

sends

message

in Serialized format

Reciver App/Consumer App

MOM/Message Broker #1

MOM/Message Broker #2

reads

the message in DeSerilized format

=> kafka do not support

MOM/Message Broker#3

kafka cluster

These MOM s/ws will

be created dynamically on demand

PTP model messaging i.e the message broker/MOM can not have Queue as the Destination

=> kafka supports only pubsub mode messaging i.e the message broker /MOM can have only Topic as the destination.

or multiple consumers

=> To send message only to one consumer.. we take the support of topic Destination. (In apache kafka) =>For one message one topic destination (Memory) is created/allocated ..that stores given data /message in partiations (huge messsage will be divied into parts/packets). Every paritiation of the message is indentified with an index like 0,1,2,3.. These indexes are technically called as offsets.

=> In kafka messaging, For one consimer one message broker instance will be allocated at a time.

So to send 1 message to 5 consumers we need

n

1 Sender ---> 1 message --> 1 Topic with sections/partiations --> 5 message broker instanaces -- 5 cosumers

=> At a time, the Message broker reads one paritiation data, takes the data, replicates data (cloned data) and sends to consumer i.e each large message/data will be stored in multiple partiations of topic and the messge broker reads data from all partiations and sends to its respective consumer.

Sender/Producer

ABC (big message)

(message)

Kafkar s/w (Kafka eco system)

Zookeeper (Boot strap server)

kafka cluster

Topic Memory

Topic with sections

message broke

message

A B C

(Topic-A)

Topic with sections

(Topic-B)

consumer#2

consumer#3

000

consumer#1

message

broker#2

message broker #3

Topic sections= topic with

Topic section

index/offset/ Topic name

id

partiations

=>Each message broker instance is capable of reading message from multiple sections/partiations of Topic destination... replicates the message parts -- merges the message parts --sends the message to respective consumer.

Apache Kafka terminologies

Message Broker /MOM :: The mediator /Middle man that will transfer message to

consumer by reading message/data from Topic /Topic partiotions

kafka cluster :: collection/group of message broker instances which are created based on cosumers count is called kafka cluster.

Topic: The memory that holds message sent by the Producer in parts

Producer :: The App/comp that sends the message to Topic Memory in Serialized format Consumer :: The App/comp that reads the message from Topic Memory through Message Broker

in DeSerialized format

offset/index :: The id or index given to partiation message/data in Topic

Zookeeper/BootstapServer :: Handles Message Broker Cluster and Topic Memory .. While managing the Message broker cluster it will do Load Balancing.

Replica-Factor ::

The no.of cloned/duplicate messages that should be created in order to send the messages to Consumers through message brokers.. Generally it is decided based consumer count.

=>if there are 10 cosumers then Replica-factor is 1/10 (For 1 message 10 clonnded copies are required)

(a)

Sender/Producer

ABC (big message)

(b) (message)

Apache kafka Work Flow Diagram

Topic Memory

Kafka. s/w (Kafka eco system) (d)

Zookeeper (Boot strap server)

kafka cluster message broker#?

message broker#2

Deperilized

Jg]

(i)

mannex

message

broker #3

consumer#3

Topic sections= topic with

partiations

0.00

consumer#1

consumer#2

(c)

message (serialized format)

(e) Topic with sections

A BC

(Topic-A)

Topic with sections

(Topic-B)

Topic section

index/offset/ Topic name

id

=>Each message broker instance is capable of reading message from multiple sections/partiations of

Topic destinations... replicates the message parts -- merges the message parts --sends the message to respective consumer.

(a) Define one Producer App

(b) Create either Static message or dynamic message that can be given at runtime

(c) Send message from Producer in the form fo key-value pair format

here "key" is topic name (With the given name topic available then uses it

"value" is the message/Data

other wise new topic will be created)

note:: Producer App sends this data/message in Serilized format.

(d) Start zoo keeper (Boot starp server)

(e) Topic Section will be created in Topic Memory to read the message sent by the producer App..

note:

New Topic section will be created (if not already available) otherwise it will use the existing Topicsection

(f) Topic section reads message and stores message in different partiations of Topic which will be created as needed.

(note:: These partiations are identified with indexes /offsets)

(g) creates link b/w Topic section/Topic and Message broker

(h) Define

one or more consumer Apps as needed by specifying topic name

(note:: Based on the given Topic name in cosumer App(s) the link b/w Topic, message broker and Cosumer App will be created as needed)

(note:: The zookeeper takes the responsbility of creating messagebroker instances as cluster based on the Consumer count and takes care of LB -Load Balancing)

(i) The Message Broker(s) reads the data/messages from Topic Partiations creates cloned/replicate copies of these partitaion messages.

partiation

j) The message broker(s) sends the cloned message to one or more consumer(s) part by part.

Arranging Apache Kafka Software

Go to https://kafka.apache.org/downloads

THE

S

3.7.0

Released Feb 27, 2024

• Release Notes

Docker image: apache/kafka:3.7.0.

• Source download: kafka-3.7.0-src.tgz (asc, sha512)

.Binary downloads:

• Scala 2.12-kafka_2.12-3.7.0.tgz (asc, sha512)

• Scala 2.13 -kafka_2.13-3.7.0.tgz (asc, sha512)

APACHE

SOFTWARE FOUNDATION

ESTABLISHED 1999

COMMU

Projects▾

We suggest the following site for your download:

https://dlcdn.apache.org/kafka/3.1.0/kafka_2.13-3.1.0.tgz

gives kafka_2.13-3.1.0.tgz

==> Copy this file (E: drive) and extract it

(h)

Sending Message using apache kafka infrastructure

note:: Kafka s/w setup gives commands in the fork

match files

to deal with ready made sender Apps and readymade receiver Apps

step1) start Zookeeper as Boot strap server..

```
E:\kafka_2.13-3.1.0\bin\windows>zookeeper-server-start
E:\kafka_2.13-3.1.0\config\zookeeper.properties
```

step2) start apache kakfa s/w setup

command as .bat file

(input value)

i) make sure that "C:\Windows\System32\wbem" is added PATH env.. variable (User Variable)

Mycomputer --->proeprties -----> advanced system settings --->

Environment Variables

User variables for NATARAJ

Variable

OneDrive

PATH

Edit environment variable

C:\Windows\System32\wbem

C:\Windows\System32

Value

C:\Users\NATARAJ\OneDrive

C:\Windows\System32\wbem;C:\Windows\System32;

X

New

Edit

ete

Browse...

X env.. variables ---> user vairables ->new --->

variable name :: PATH

value ::

C:\Windows\System32\wbem

ok --->ok --->ok

***

aslo add System32 folder to PATH env.. variable

note:: make sure that current dirve tmp folder's kafka folder deleted which is having log messages

```
ii) E:\kafka_2.13-3.1.0\bin\windows>kafka-server-start
E:\kafka_2.13-3.1.0\config\server.properties
```

step3) Create new topic specifying reflection factor (no.of copies), partintiations count (offset count)
E:\kafka_2.13-3.1.0\bin\windows>kafka-topics --create --bootstrap-server localhost: 9092 --replication-factor 1 --partitions 1 --topic nit-tpc

with message broker

step4) Create a producer and link

with support of bootstrap-server

E:\kafka_2.13-3.1.0\bin\windows>kafka-console-producer --bootstrap-server localhost:9092 --topic nit-tpc
hello

>hai

send these messages

messages

>123

after starting the consumer

step5) create a consumer and link with message broker

E:\kafka_2.13-3.1.0\bin\windows>kafka-console-consumer.bat --bootstrap-server

localhost:9092 --topic nit-tpc

hello

hai

123

if the Zookeeper server start is giving connection timeout exception (connection=18000 ms) then delete kafka related log files (all folders and files in d:\tmp folder)