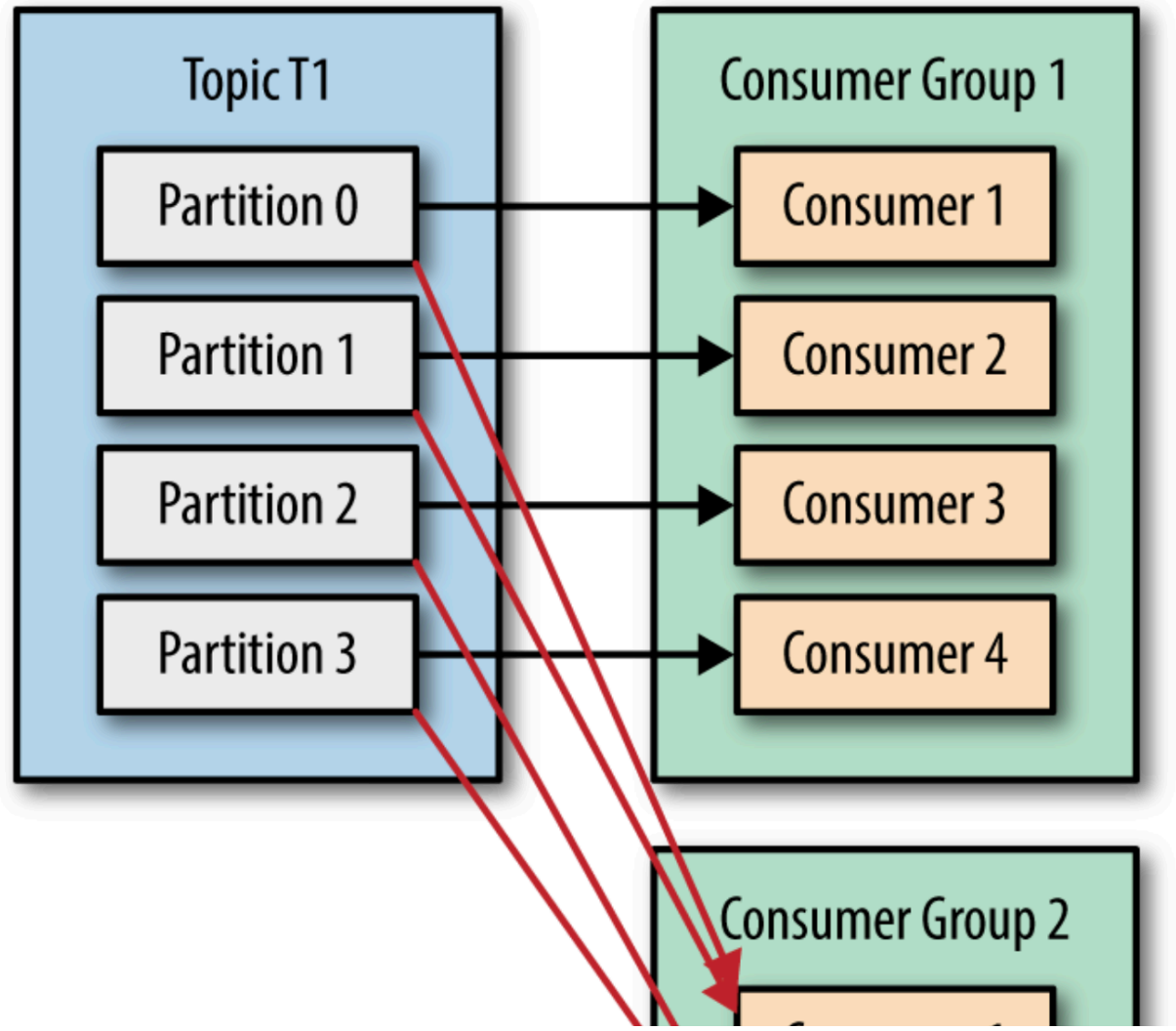
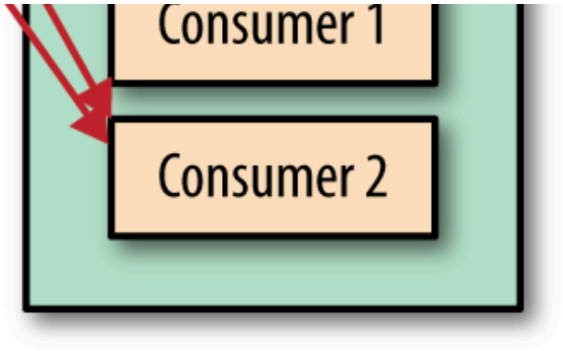
# Kafka

Wednesday, 15 May 2024 8:34 AM

- Kafka is  Organizes the messages in to Topics
- Each topic has set of partition
- Each partition is a replicated log of messages called , reference as sequential off set
- Each partition is replicated 3 times
- Each replica leaves on separate broker
- Leader handles all read and write process
- Zool keeper manages the replication of
- If you have 2 consumer with same group ID with 12 partition then e consumer will read 6 partition each
-



each




Within same group: **NO**

Pache Matching : It is some thing producer produces more messages that cannot be handle consumer then we can place messages in que so that consumer can process one by one
P2P: Here one producer is publishing the message then it will be red by one consumer then second consumer subscribed to same topic then only one consumer can read this message.

POP/SUB: Here one message can be published to multiple Q's through exchange
If partition is not defined then it will follow round robin logic to place a message in partition Example Topic has 3 partition then first message will go to partition 0 then 2 and 3 . 4th mes will come to partition 0

Consumer group is needed is because if we configure multiple consumer inside consumer gr then in any case consumer one goes down then Zookeeper will redirect to other consumer t read data from where primary consumer is left over .

We can configure the topic to retry mechanism. If that does not work after n number of retr that is been configured then it will put the message dead letter topics
Kafka uses pull back mechanism but rabit mQ uses push back mechanism

Offset explorer UI used in Kafka to view the lag information and partition information where data is stored

Branching Strategy

Kafka Streems

Below Link for
https://rathod-ajay.medium.com/kafka-interview-question-for-experienced-developer-in-2024-01b95d126cdd

Stream Processing Characteristics

1) Unbounded data sets : While processing the data sets , It is not known how many data sets are present and expected to come
2) One record at a time : Each record is analyzed , transformed

d by

.
ssage

roup
o

y

e

a

2) One record at a time : Each record is analyzed , transformed
3) Real time computation : Recorded are processed inside generics and pushed to next st
   In real
4) Low latency
5) Parallel processing

https://developer.confluent.io/courses/spring/spring-boot-with-apache-kafka-hands-on/

Apcahe kafka is is highly scalable and distributed platform for creating and processing the

Open sourced in 2011 and developed by linked in
Kafka has 3 component

1) Kafka Connect
2) Kafka streams - Real time stream processing
3) KSQL- DB
4) Kafka Broker
5) Kafka client- For producer and consumer

Topic is data stream and we can compare to database table
Kafka cluster is nothing but group computer that contain the kafka server . When ever is
each partition in to multiple server

Kafka consumer group is nothing but group of consumers but max consumer can be less
not accept if consumer is more than the partition . This is due to double rating of records

Kafka connect : is between data source and kafka cluster on source and destination , Kafk
target system database . As being the developer we don't need to write single line of cod

The source connector is called Source connecter and destination is called as Sink connect

Kafka Connect is called cluster . Each unit in the cluster is called as worker .
In Kafka connect( Source and Sink) both we can apply the transformation logic like modif
Kafka connect will self manage following activities. When any worker goes down with in
worker . In this way load will be balanced

1. Reliability
2. High Avilability
3. Scalability
4. Load Balancing

Inside the kafka connect we need to configure the JDBC connection and define the task (
the kafka connect cluster once processing is complete it task will provide the update to w

tage

e streams in real time

partition is created kafka server will distribute the data of

than or equal to number partition of the TOPIC . Kafka will
s or not to process the duplicate the records.

ka connect will read data from kafka cluster and post in to
le

tor .This is uses Kafka connect frame work

ying the records
Kafka connect cluster the task are reassigned  to other

configure the the tables ) then task will be distributed across
worker

KSQL : It is SQL interface to Kafka Stream . It has below 2 modes

1)Interactive mode- command line interface or Web interface we can submit the query .

2) 2.Headless Mode: Here we can submit the KSQL file which are executed by KSQL server .

KSQL comes with 3 components

3) KSQL Engine
4) REST Interface
5) KSQL Client(CLI/UI)

Combination of KSQ Engine and Rest interface form the KSQL Server . It can be deployed

Using KSQL you can group , apply filter , join topics & aggregate the topics

Sink the result of your query to another topic

**Kafka comes with 3 different flavor**

1) Open Source - Aache kafka
2) Commercial Distribution - confluent .io - Used in prod , it is also managed service
3) Managed service -confluent, amazon, aiven.io - U don't need to down load the software provider

From confluent.io website and hit on down load button

To start the kafka zookeepr

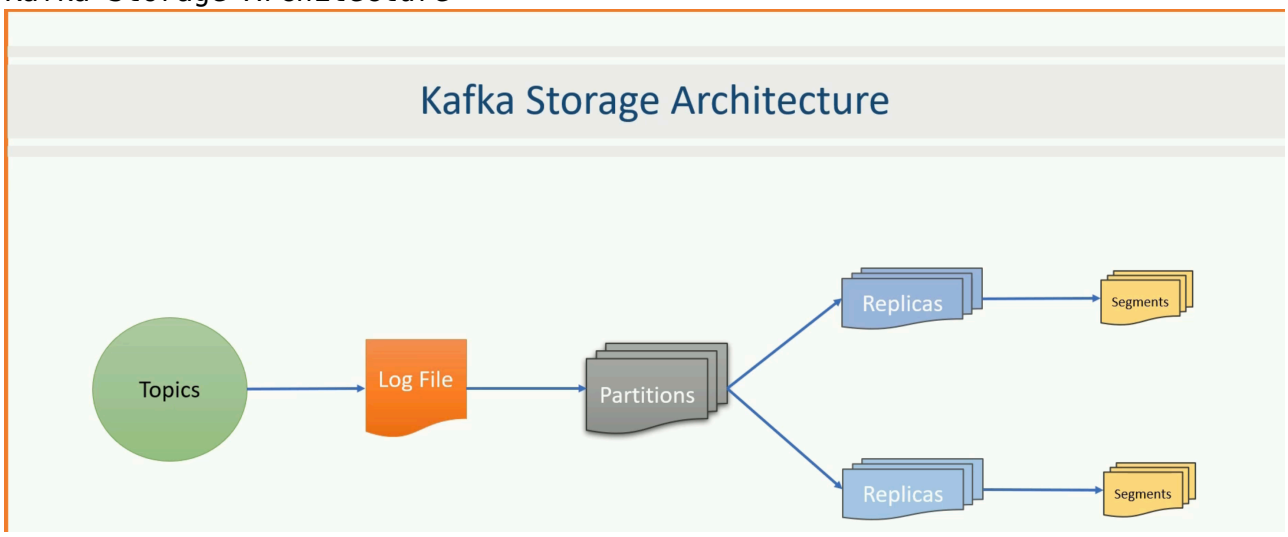
`sh zookeeper-server-start /etc/kafka/zookeeper.properties`

Below are the properties to start multiple instance of kafka (kafka cluster)

Under this path /Downloads/confluent-7.6.1/etc/kafka you can find server. properties yo

- Broker-id =0 by default you can change it to 1 , 2 and so on but in real time you ma separate machine
- Lister port you need to change by default it is `#listeners=PLAINTEXT://:9092 unco`
- `Logfile name you need to change log file log.dirs=/tmp/kafka-logs you can`

`Kafka Storage Architecture`



Good for development and test env
Good for prod env

in interactive mode or headless mode

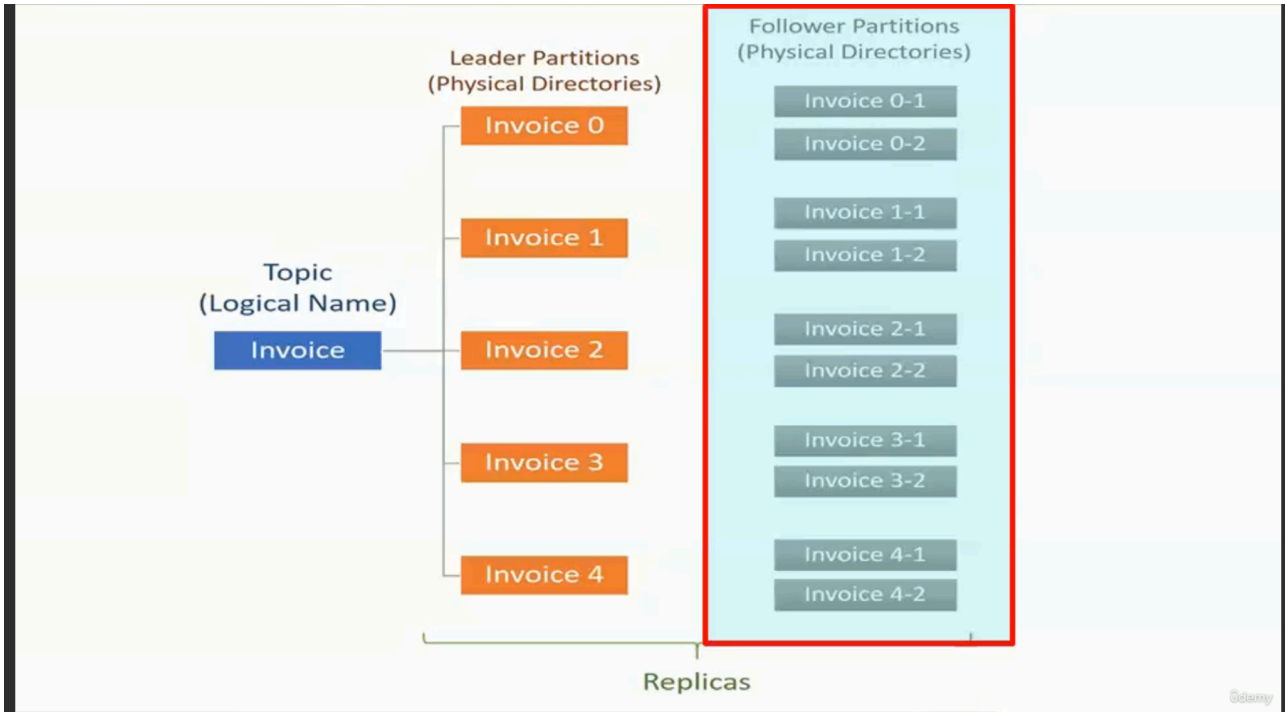
and manage the cluster . All these will be managed by cloud

ou need to make following changes
y not need to do it because each server is started on

```
mment and change respectively
change to log.dirs=/tmp/kafka-logs1 etc
```

Topic will be divided in to partition example million messages are not able to keep at sing
partition
Replication  factor is nothing but how many copies you need to keep at each partition. Fo
partition and 3 replicas then 15 directory should be created . In this example 5 directory
servers

We can categorize the  topic partition  replicas in to 2 category
Leader partition - each partition created during topic is leader
Follower partition - each replica set created during topic creation is follower

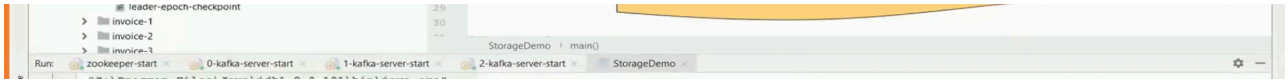



Off set message is of 64 bit integer giving the unique ID for each message . In case off rea
next offset id from last segment . You can see in below screen shot initial .log file is starte

gle place and divided and placed at multiple place called

or example while creating the topic you have specified the 5
on each broker will be created in case if you started 3 Kafka

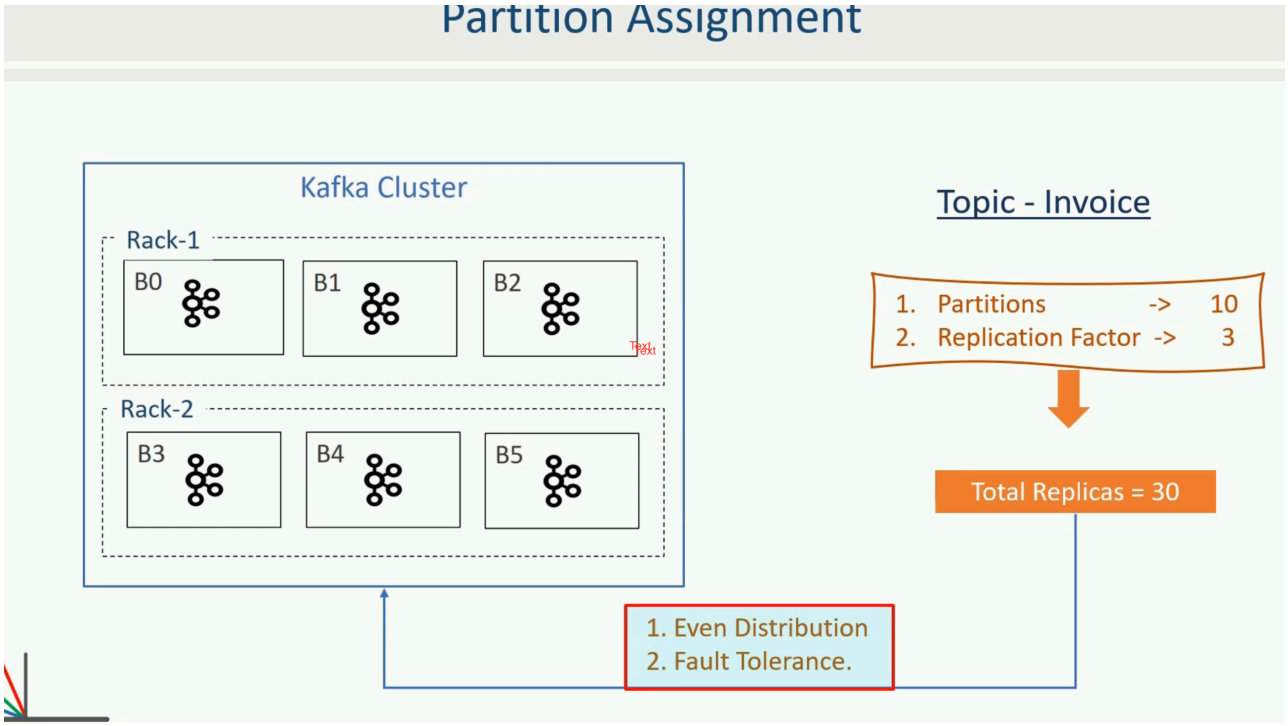
aches the max size then new segment will be created with
ed with zero and next log file with some numbers



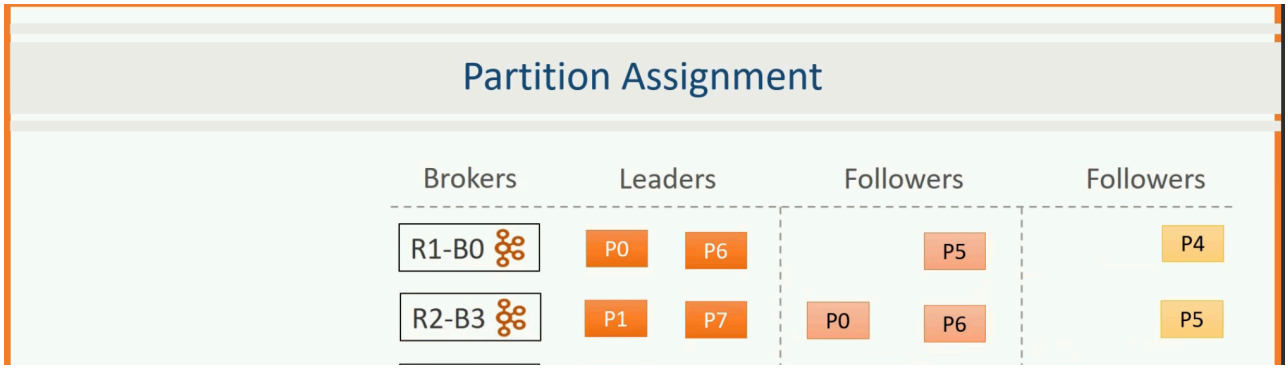
In the above screen shot .index will main the off set index of messages and .timeindex wi
message from certain time

Kafka Cluster is nothing but group of brokers created and  work tougher to perform the t
We can create 100 plus broker in single cluster .
Controller is managing the brokers in the cluster , If any broker dies or left cluster it will r
become the lead broker

Partition assignment is show as per the below diagram , For example if the Kafka Cluster
If we are creating the one topic with 10 partition with the replication factor of 3 . Total pa
sure scalability  will be achieved and copy of the request will be saved in different machir




This is how Brokers, leader and followers are distributed in the cluster . First Brokers are
Then leaders ae assigned to each broker . Finally followers will be assigned in the fashion



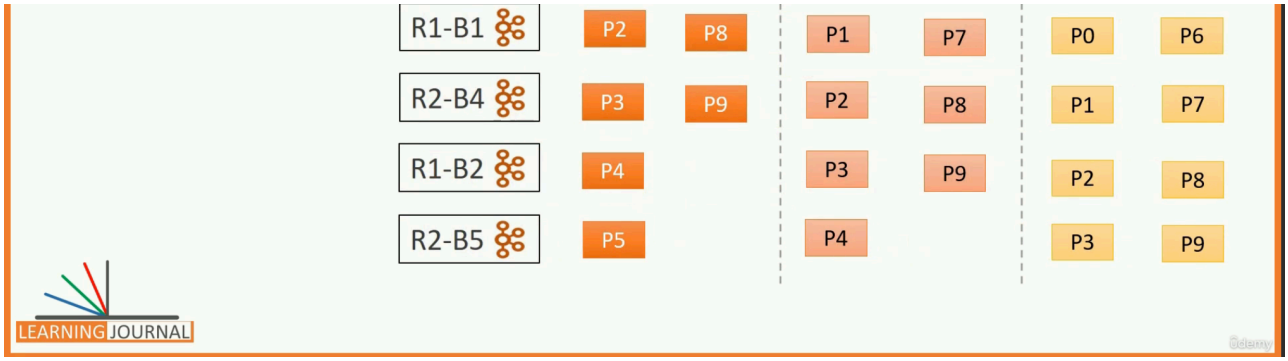
ll maintain the time . In case consumer wants to read

ask

eassign the work to other broker . So that other broker

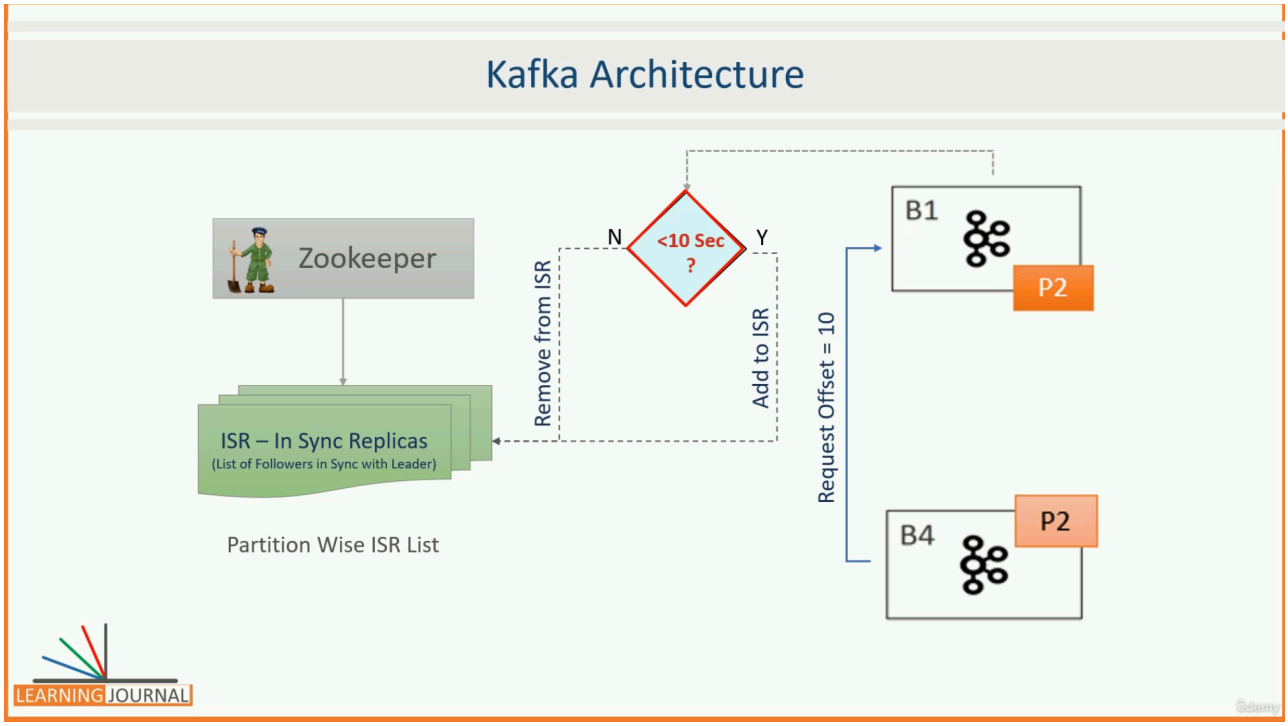
is  created with 6 broker and placed on 2 racks .
artition is 30 will be distributed even as possible to make
ne to make sure to achieve the fault tolerance

arranged in order like one broker from one rock .
 of leaving first broker




How does Producer make a request to Topic.
Producer will query for broker in Kafka cluster and it will respond back with meta data wi
Producer will decide which partition it needs to send data . On receiving end also it will re
Followers will not accept message from producer or consumer . Followers will request th

ISR List-In Sync Replica : ISR refers to the replicas of a partition that are “in sync” wi
As soon as follower request for message ,it will send from lead broker . First time followe
will be from 10 onwards. So that leader will understand the message from 0 to 9 was per
In this scenario there some time needed to sync between leader and broker that time by
configuration . In sync replicas contain the followers who are sink with leader with in con




Committed vs uncommitted message: The message which is successfully sent to follower
called un committed message.
Minimum In Sync replicas : It is managed by kafka incase Kafka brokers went down for so
will exists only in leader. It is a risky situation in order to avoid this we need to set min-in
The negative thing is if we set the min-insync-replicas to 2 then until 2 replicas commit th
read only mode.

ith all leaders .
ead from Leader Broker.
e message and leader will send a message to stay in sink

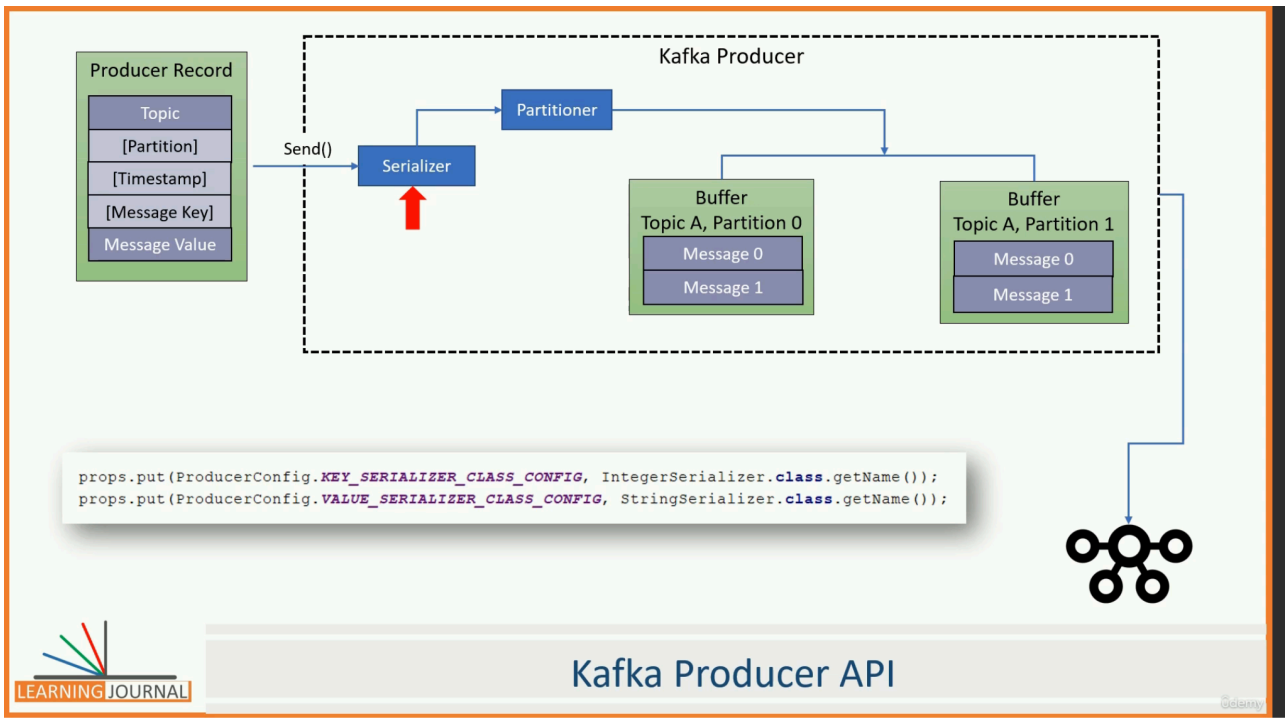
th the leader.
r will make request from off set 0 to 9 . Then next request
sist .
default is 10 sec . You can increase or decrease using Kafka
figured seconds .(This scenario it is 10 sec).

is called committed message. Which is not in follower is

me reason and left with only Leader broker . Then our data
sync-replicas : 2.
e message . Topic will not allow to write message and it is in

Kafka Producrer API

In producer API there are 5 parameters , 2 are mandatory ( topic nam
partition) .Below is the producer architecture . Before the message ne
serialized .For complex serialization you can use avro , json etc




Kafka broker has multiple partition . In order to send the message to s
parameter while sending the request .
Second option is we need to specify the key in the request then Kafka
request needs to be placed .
By default kafka has default partitioner (Hash Key Partitioning , Round
Kafka uses the key and find the hash code associated with key and ide
messages goes to same partition . This algorithm takes total number o
increase the partition then default partitioner will start giving differen
initially create enough partition or 25 % more . The disadvantage is if
need to re publish the messages.

In case if the key is not specified then default partitioner will use rou
each broker
Once message is reached the Buffer , I/O thread will transfer the mess

e and message) optionals are ( key, timestamp ,
eeds to be send to broker it needs to be

specific partition you can specify the partition

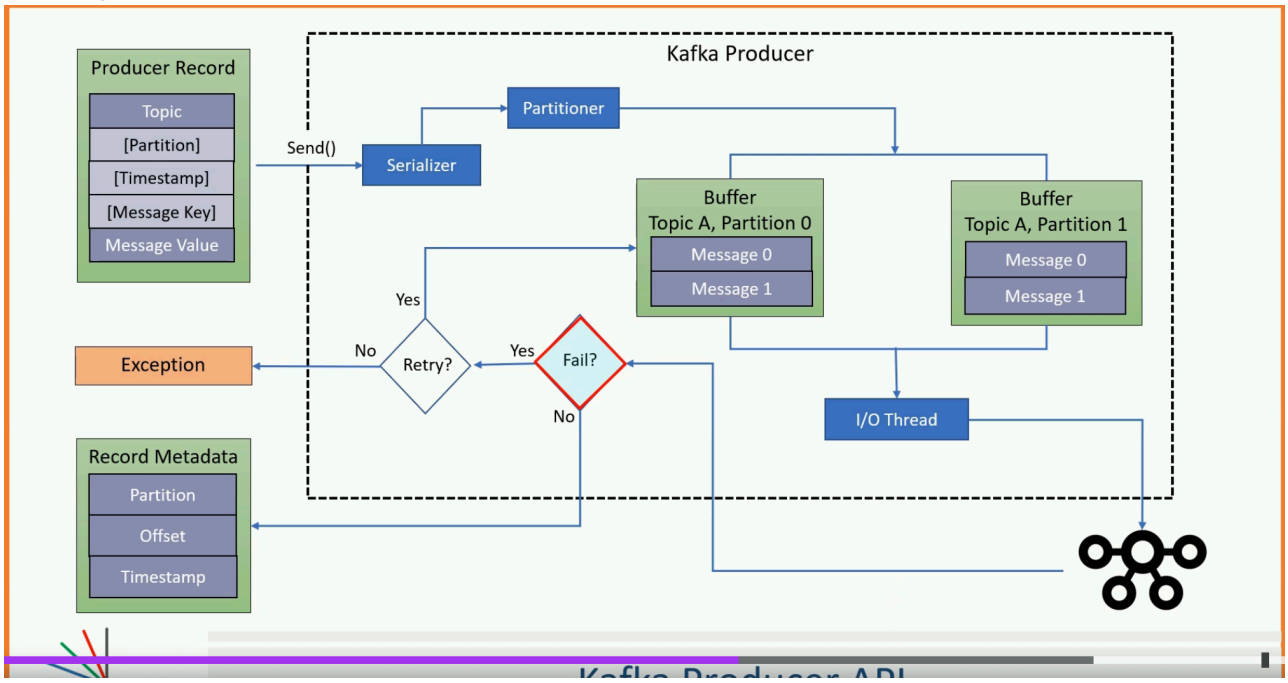
will use the key to find the partition where

l robin Partitioning) .
entify partition . Hashing ensures same key
of partition as another argument . If you
t out put. Means while creating the partition
you increase the partition drastically then you

nd robin process to distribute the messages to

sage to topic . If buffer memory is exhausted

then send method will get I/O exception in that case we need to incre
Memory in configuration class . Default memory size is 32 mb.
Complete flow is below




At leastonce & At Most Once

At leastonce: Once producer send the message it will be stored in kafk
success acknowledgment from broker . IO will try to resend a message
but it will not reach the IO due to network error . In this case produce
network error . In this scenario Kafka does not have mechanism to ide
same message to log file . This implementation is knows as atleast onc

Atmost once means can be achieved by setting the retry to zero . In th
have duplicate.

We need to set this one as part of properties configuration in produce

**Producer Configuration:**

java Copy code

```
Properties props = new Properties();
props.put("bootstrap.servers", "localhost:9092");
props.put("key.serializer", "org.apache.kafka.common.serialization.StringSeriali
props.put("value.serializer", "org.apache.kafka.common.serialization.StringSeria
props.put("acks", "0");  // No acknowledgment
props.put("retries", "0");  // No retries
props.put("delivery.timeout.ms", "30000");  // Maximum wait time for acknowledgm
KafkaProducer<String, String> producer = new KafkaProducer<>(props);
```

ase the memory of Producer using buffer.

ka log file and incase IO thread did not get the
e this time broker sends success message to IO
r will try to resend the same message due to
entify the duplicate entry it and it will store
ce
his scenario you many loose messages but never

er API . Below is the example.

**Consumer Configuration:**

```java
Properties props = new Properties();
props.put("bootstrap.servers", "localhost:9092");
props.put("group.id", "my-group");
props.put("key.deserializer", "org.apache.kafka.common.serialization.StringDeser
props.put("value.deserializer", "org.apache.kafka.common.serialization.StringDes
props.put("enable.auto.commit", "true");  // Auto commit offsets
props.put("auto.commit.interval.ms", "5000");  // Commit interval
KafkaConsumer<String, String> consumer = new KafkaConsumer<>(props);
```

Exactly Once : By enabling the property called enable.idempotence = t
implementing exactly once
Initially when producer is handshake with broker. Producer Api will as
producer ID for each producer.
Next step is producer API will start assign the unique message sequen
send message it will uniquely identify the message using producer ID a

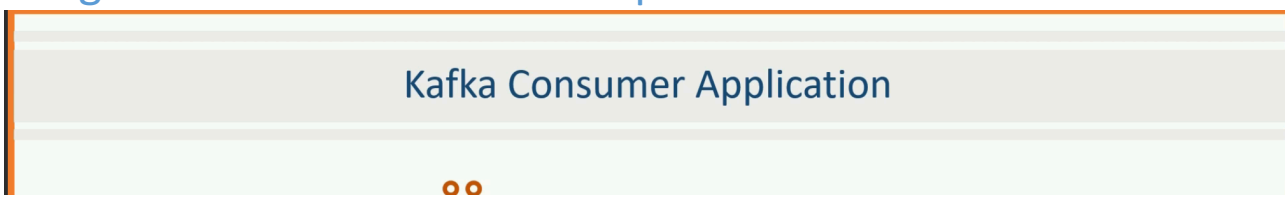
POS(Point of Sale) simulator in Kafka: It basically helps us to send milli
arguments

- Topic name: Name of the topic where we need to send message
- Number of Producer thread: How many parallel thread needs to
- Produce speed: Where we can specify the time interval between

Note: Size of the message we cannot specify that is the draw back
Need to do practical practise

Kafka Consumer Example architecture

We need to define consumer group that contains the multiple consum
same topic then kafka will take care of scalability and fault tolerance.
per second for 10 topic then target consumer group has 5 consumer t
consumer by Kafka . In the scenario like 10 consumer and 10 partitior
assign the current consumer topic to different consumer.



true. If we enable this kafka take care of

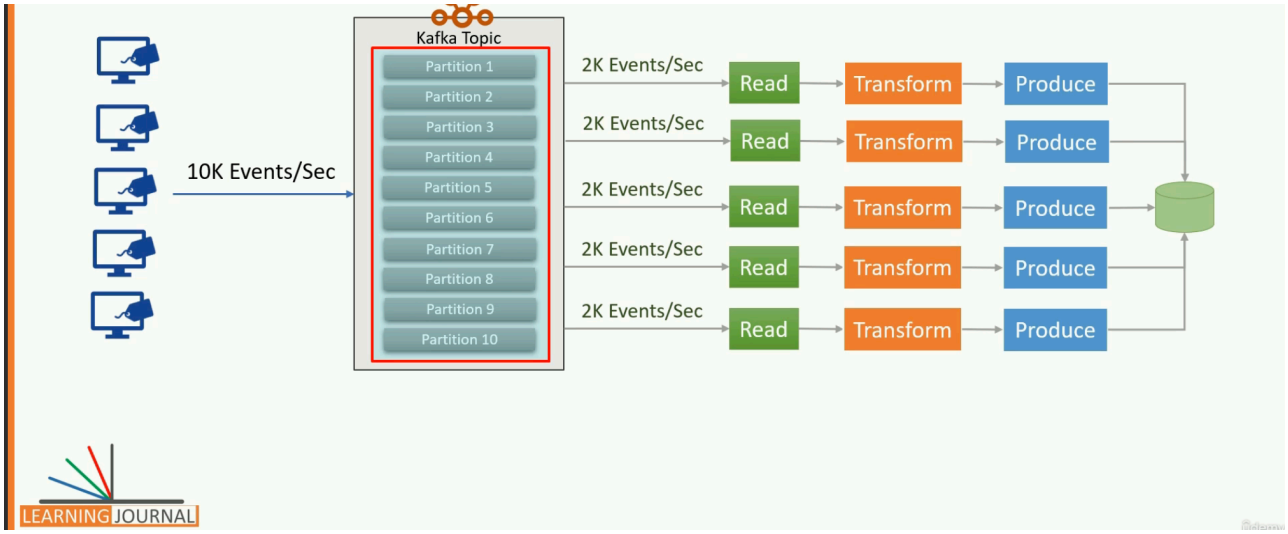
k for ID and broker will  assign the dynamic

ce number . It starts from zero.When IO thred
and sequence number

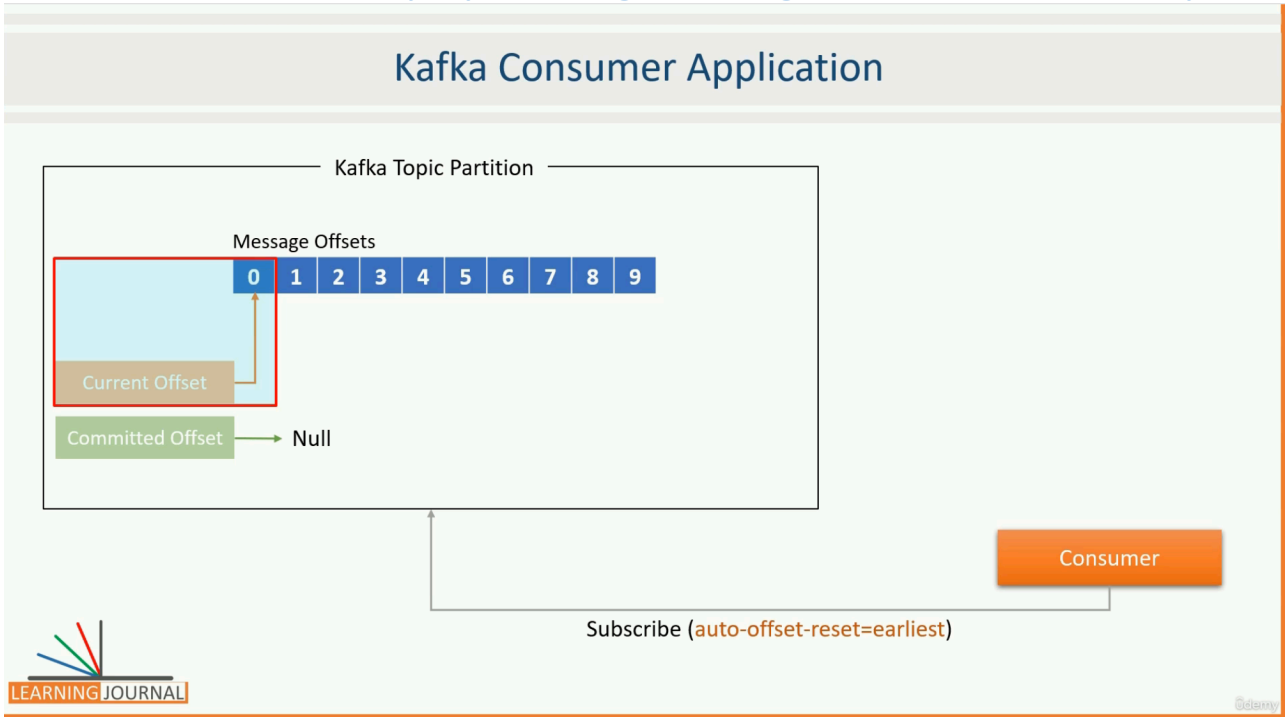
ons of messages in parallel. It takes 3 major

be enabled to send message
the messages.

ners. When consumer group is subscribed to
In the below picture if 10k events are produced
hen 2k messages will be distributed for each
n . If any consumer goes down then Kafka will re




Kafka maintains the unique records using the offset ID & also current
partition. Initially the current offset will be null . It means consumer n
You can set the property auto-offset-reset=earliest. It means current
means will send only upcoming messages after the subscription.




Once consumer start reading the messages the offset will be advance
offset will be set to 4 . In case of consumer restarts the offset will rese
again that is duplicate. To avoid that situation kafka maintains the cor
down then other consumer can read data by using commit offset num
Note: The committed offset will be updated at broker only after consu
data base or different mS while going to poll(read) next message .

offset and committed offset with in the
eeds to start reading from offset number 0 .
offset is set to zero. The default value latest that

d . Example if 3 messages are consumed then
et and it will result in getting same message
mmitted offset . It will help if the consumer goes
nber as it is securely stored with broker.
umer is successfully read and update the target

Learn : https://www.linkedin.com/in/prashant-kumar-pandey/