**Day\_6**

3 node Kafka cluster setup.

**Content:**

Kafka cluster setup with 3 Kafka nodes.

Prerequisites:

* Java-8 or higher.
* Min 1 GB RAM required.
* Kafka needs zookeeper node to start, hence we need one or more zookeeper nodes in running state.

**Kafka Installation:**

* Go to Kafka’s official download page.

(<https://kafka.apache.org/downloads>)

* Download the latest stable release:

([kafka\_2.13-3.4.0.tgz](https://downloads.apache.org/kafka/3.4.0/kafka_2.13-3.4.0.tgz))

Scala version: 2.13

Kafka version: 3.4.0

* Now create three separate directories and copy the downloaded binaries in it.
* Extract the files.
* Go to /kafka\_2.13-3.4.0/config/server.properties and make below changes in the file:

broker.id=**1**

listeners=PLAINTEXT://**localhost:9092**

num.network.threads=3

num.io.threads=8

socket.send.buffer.bytes=102400

socket.receive.buffer.bytes=102400

socket.request.max.bytes=104857600

log.dirs=**/tmp/kafka-logs-1**

num.partitions=1

num.recovery.threads.per.data.dir=1

offsets.topic.replication.factor=1

transaction.state.log.replication.factor=1

transaction.state.log.min.isr=1

log.retention.hours=168

log.retention.check.interval.ms=300000

zookeeper.connect=**localhost:2181,localhost:2182,localhost:2183**

zookeeper.connection.timeout.ms=18000

group.initial.rebalance.delay.ms=0

We have updated highlighted values only from already existing file.

Below is the explanation of each updated field:

* **broker.id=1 :** This is unique broker id which should be unique for all the nodes. (for node 2 it will be 2 and for node 3 it will be 3 and so on.)
* **listeners=PLAINTEXT://localhost:9092 :** This node will now listen on 9092 port. (For node 2 and node 3 ports will be unique. e.g. 9093, 9094)
* **log.dirs=/tmp/kafka-logs-1 :** In this directory all the logs and brokers information will be saved. (for other nodes we will change directory names to kafka-logs-2 and kafka-logs-3)
* **zookeeper.connect=localhost:2181,localhost:2182,localhost:2183 :**

This is URL and port, configured to connect to the zookeeper servers.

We have mentioned all the node’s information here, as if it will fail to connect to one server it will try to connect to next server.

* We are ready with setup now.
* Start our zookeeper servers.
* Now we can start all the kafka broker’s one by one, using below command:

cd kafka\_2.13-3.4.0/

bin/ kafka-server-start.sh config/server.properties

* Now we can check if our brokers are running or not and other information like leader and follower using below command:

Echo dump | nc localhost 2181 | grep broker

**Output:**

**/brokers/ids/1**

**/brokers/ids/2**

**/brokers/ids/3**

* Now we can work on our brokers.
* We can create topics using all 3 brokers.

e.g.

./kafka-topics.sh --bootstrap-server localhost:9092, localhost:9093, localhost 9094 --create --topic Topic-2 --partitions 3 --replication-factor 3

* Now we can check if topic is created or not using below command:

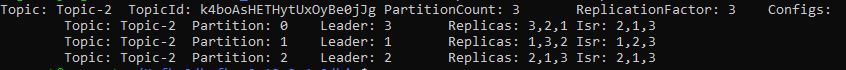
./kafka-topics.sh --bootstrap-server localhost:9092 –list

We can use any port from our three ports from 9092, 9093, 9094.

* Describe the topic using below command:

./kafka-topics.sh –bootstrap-server localhost:9092 –describe –topic Topic-2

**Output:**



* In above result we can see ISR which is In Sync Replicas. It shows that all replicas are in sync. If we will stop one server, we can see that stopped broker id will get removed from the Isr field.
* Also, when we will stop one server that time leader of the partition will get changed and it will be another broker now.
* And even we will start that node still leader will be the one which was assigned when we stopped our broker. But Isr will get changed again