# Kafka 3 node cluster setup

1. Java is required for kafka. If not already present, install it using  
    **$ *yum install java***   
    **$ *yum install java-devel***
2. Download kafka (latest version 2.5.0) binary using the command   
    ***$ curl “http://apache.spinellicreations.com/kafka/2.5.0/kafka\_2.12-2.5.0.tgz” -o kafka.tgz***    
     
   For the older version (0.11.0.3)  
    **$ *curl “https://archive.apache.org/dist/kafka/0.11.0.3/kafka\_2.11-0.11.0.3.tgz” -o kafka.tgz***
3. Extract the files to a directory  
   **$ *tar –xvzf kafka.tgz --strip 1***
4. For zookeeper setup, you can use the zookeeper which comes along with the kafka package or you can separately download the zookeeper package. For using the zookeeper which is in the kafka package follow the below steps and repeat on every node in the cluster.  
   1. Define the configuration for the zookeeper in the ***kafka/config/zookeeper.properties*** file by defining the following configuration parameters -   
       ***tickTime=2000***  
       ***initLimit=10***  
       ***syncLimit=5***  
       ***dataDir=<path for data directory>***  
       ***dataLogDir=<path for log data directory (if not defined, then datadir will be used)>***  
       ***clientPort=2181***  
       ***server.1=<node 1 address>:2888:3888***  
       ***server.2=<node 2 address>:2888:3888***  
       ***server.3=<node 3 address>:2888:3888***  
       ***autopurge.snapRetainCount=3***  
       ***autopurge.purgeInterval=24***  
        
      The details for the configuration parameters can be found [here](https://zookeeper.apache.org/doc/current/zookeeperStarted.html).
   2. In the <***datadir***> folder add a file ***myid*** and add the node id 1 to the file in the first node. (This must be a single integer value).  
      Similarly, for nodes 2 and 3, add their respective ids in <***datadir***>***/myid*** file on the respective nodes.
   3. From the kafka directory, start the zookeeper server using the config defined above   
       **$ *./bin/zookeeper-server-start.sh config/zookeeper.properties***   
        
      For independent zookeeper installation and setup follow the steps listed in Zookeeper installation and setup
5. Update the kafka server configuration in the ***config/server.properties*** file in the kafka directory as follows -   
   1. Define a unique broker id for each kafka server.  
       ***broker.id=0***   
      Note : It is possible to have multiple kafka server instances on a single node. In that case we need to define separate ***server.properties*** file for each instance.
   2. Define a directory for storing of log files  
       ***log.dirs=<path for storing logs>***  
      Note : It is possible to define a comma separated list of directories
   3. To form a cluster of 3 nodes, add a comma separated list of node and port addresses in the ***zookeeper.connect*** parameter so that if a zookeeper instance fails, the node will automatically try to connect to the next available address  
       ***zookeeper.connect= <node 1 address>:2181,***  
       ***<node 2 address>:2181,***  
       ***<node 3 address>:2181***
   4. Repeat the above steps for each node in the cluster.
6. From the kafka directory run the kafka server on each node  
    ***$ ./bin/kafka-server-start.sh config/server.properties***
7. Now, run the following command on any one node  
    **$ *./bin/connect-distributed.sh config/connect-distributed.properties***
8. On one of the nodes create a topic named test  
    **$ *./bin/kafka-topics.sh --create –bootstrap-server <list of server:port> --replication-factor 3 --partitions 1 --topic <topicname>***  
     
   Note: Here ***<list of server:port>*** can be ***localhost:9092*** as defined in the config/connect-distributed.properties file  
   If using the older version, use ***-- zookeeper*** instead of ***--bootstrap-server*** .  
     
   Verify it by   
    **$ *./bin/kafka-topics.sh --list –bootstrap-server <list of server:port>***
9. On the same node run the producer script to publish a message  
    **$ *./bin/kafka-console-producer.sh --bootstrap-server <list of server:port> --topic test***  
    ***> This is a message***  
    ***> This is another message***  
     
   Note: Here ***<list of server:port>*** can be ***localhost:9092*** as defined in the config/connect-distributed.properties file  
   If using the older version, use –***broker-list*** instead of ***-- bootstrap-server*** .
10. On other nodes run the consumer scripts to read the message from the beginning  
     **$ *./bin/kafka-console-consumer.sh --bootstrap-server <list of server:port> --topic test --from-beginning***  
     ***> Test Msg1***  
     ***> Test Msg2***   
      
    Note: Here ***<list of server:port>*** can be ***localhost:9092*** as defined in the config/connect-distributed.properties file

# Zookeeper installation and setup

1. Download zookeeper using the command   
    **$ *curl “https://mirrors.sonic.net/apache/zookeeper/zookeeper-3.6.1/apache-zookeeper-3.6.1-bin.tar.gz” -o zookeeper.tar.gz***
2. Extract the files to a directory  
    **$ *tar –xvzf zookeeper.tar.gz --strip 1***
3. Add a ***zoo.cfg*** file to the ***conf*** directory and add the following lines to the file.  
     
    ***tickTime=2000***  
    ***initLimit=10***  
    ***syncLimit=5***  
    ***dataDir=<path for data directory>***  
    ***dataLogDir=<path for log data directory (if not defined, then datadir will be used)>***  
    ***clientPort=2181***  
    ***server.1=<node 1 address>:2888:3888***  
    ***server.2=<node 2 address>:2888:3888***  
    ***server.3=<node 3 address>:2888:3888***  
    ***autopurge.snapRetainCount=3***  
    ***autopurge.purgeInterval=24***
4. From the zookeeper directory, start the zookeeper server  
    $ ./bin/zkServer.sh start

Using the python Client for kafka

1. Install python 3.6  
    **$ *yum install python36***
2. Download confluent kafka client for python  
    **$ *pip3 install confluent-kafka***
3. For running a producer client, run the following code snippet  
     
    ***from confluent\_kafka import Producer***  
     
    ***p = Producer({'bootstrap.servers': 'localhost:9092'})***  
    ***def delivery\_report(err, msg):***  
    ***""" Called once for each message produced to indicate delivery result.***  
    ***Triggered by poll() or flush(). """***  
    ***if err is not None:***  
    ***print('Message delivery failed: {}'.format(err))***  
    ***else:***  
    ***print('Message delivered to {} [{}]'.format(msg.topic(), msg.partition()))***  
     
    ***for loop in range(10):***  
    ***p.produce('test', 'This is test msg '+str(loop), callback=delivery\_report)***  
     
    ***p.flush()***  
     
   The above code will publish 10 messages to the topic **test**  
   Note : the topic must already be created on the broker
4. For running a consumer client, run the following code snippet  
     
    ***from confluent\_kafka import Consumer***  
     
    ***c = Consumer({***  
    ***'bootstrap.servers': 'localhost:9092',***  
    ***'group.id': 'mygrp',***  
    ***})***  
    ***c.subscribe(['test'])***  
     
    ***while True:***  
    ***msg = c.poll(1.0)***  
    ***if msg is None:***  
    ***continue***  
    ***if msg.error():***  
    ***print("Consumer error: {}".format(msg.error()))***  
    ***continue***  
    ***print('Received message: {}'.format(msg.value().decode('utf-8')))***  
     
    ***c.close()***  
     
   The above code will subscribe to the topic **test**  
   Note : the topic must already be created on the broker