How to Install Apache Kafka (Single Node) on Ubuntu and Debian

Apache Kafka is a distributed streaming platform. It is useful for building real-time streaming data pipelines to get data between the systems or applications. Another useful feature is real-time streaming applications that can transform streams of data or react on a stream of data. This tutorial will help you to install Apache Kafka on Ubuntu and Debian systems.

Step 1 – Prerequisites

Apache Kafka required Java to run. You must have java installed on your system. Execute below command to install default OpenJDK on your system from the official PPA's. You can also install the specific version of from here.

```
sudo apt update

sudo apt install default-jdk
```

Step 2 – Download Apache Kafka

Download the Apache Kafka binary files from its official download website. You can also select any nearby mirror to download.

```
wget http://www-us.apache.org/dist/kafka/2.2.1/kafka_2.12-2.2.1.tgz
```

Then extract the archive file

```
tar xzf kafka_2.12-2.2.1.tgz

mv kafka_2.12-2.2.1 /usr/local/kafka
```

Step 3 – Start Kafka Server

Kafka uses ZooKeeper, so first, start a ZooKeeper server on your system. You can use the script available with Kafka to get start single-node ZooKeeper instance.

```
$ cd /usr/local/kafka

hadoop@ubuntudev:/usr/local/kafka$

bin/zookeeper-server-start.sh config/zookeeper.properties
```

Now start the Kafka server:

```
hadoop@ubuntudev:/usr/local/kafka$

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

...

[2018-03-13 10:47:45,989] INFO Kafka version : 2.2.1
(org.apache.kafka.common.utils.AppInfoParser)
```

```
[2018-03-13 10:47:45,995] INFO Kafka commitId : c0518aa65f25317e
(org.apache.kafka.common.utils.AppInfoParser)

[2018-03-13 10:47:46,006] INFO [KafkaServer id=0] started
(kafka.server.KafkaServer)
```

Step 4 – Create a Topic in Kafka

Let's create a topic named "testTopic" with a single partition and only one replica:

```
hadoop@ubuntudev:/usr/local/kafka$

bin/kafka-topics.sh --create --zookeeper localhost:2181 --replication-
factor 1 --partitions 1 --topic testTopic

Created topic "testTopic".
```

The replication-factor describes how many copies of data will be created. As we are running with single instance keep this value 1.

Set the partitions options as the number of brokers you want your data to be split between. As we are running with a single broker keep this value 1.

Now you can see the created topic on Kafka by running the list topic command:

```
hadoop@ubuntudev:/usr/local/kafka$

bin/kafka-topics.sh --list --zookeeper localhost:2181
```

```
testTopic
```

Alternatively, instead of manually creating topics you can also configure your brokers to auto-create topics when a non-existent topic is published to.

Step 5 – Send Messages to Kafka using Producer

The "**producer**" is the process responsible for put data into our Kafka. The Kafka comes with a command line client that will take input from a file or from standard input and send it out as messages to the Kafka cluster. The default Kafka send each line as a separate message.

Let's run the producer and then type a few messages into the console to send to the server.

```
hadoop@ubuntudev:/usr/local/kafka$

bin/kafka-console-producer.sh --broker-list localhost:9092 --topic
testTopic

>Welcome to kafka

>This is my first topic

>

Ctrl C to exit
```

You can exit this command or keep this terminal running for further testing. Now open a new terminal to the Kafka consumer process on next step.

Step 6 – Using Kafka Consumer

Kafka also has a command line consumer to read data from Kafka cluster and display messages to standard output.

```
bin/kafka-console-consumer.sh --bootstrap-server localhost:9092 --topic
testTopic --from-beginning



Welcome to kafka

This is my first topic
```

Now, If you have still running Kafka producer (Step #5) in another terminal. Just type some text on that producer terminal. it will immediately visible on consumer terminal. See the below screenshot of Kafka producer and consumer in working: