howtoprogram 5:46 am on January 5, 2018

Tags: Apache Kafka (11), Big Data (3)

Getting Started With Apache Kafka 1.0

In this article, I'd like share some basic information about **Apache Kafka**, how to install and use basic client tools ship with Kafka to create topic, to produce/to consume the messages. The version used is **Apache Kafka 1.0.0**.

1. Concept

Apache Kafka is a Publish-Subscribe messaging system, originated at Linkedin in 2009, open sourced in 2011. In term of architecture, Apache Kafka is similar to other messaging systems like ActiveMQ, RabbitMQ, …However, Apache Kafka has some characteristics that make it more popular nowadays such as:

- Kafka was designed to be distributed inherently. This makes Kakfa be very easy to scale out.
- High throughput, high performance
- Guarantee the fault-tolerant in term of machine failure.

Apache Kafka now becomes very popular for use cases related to real-time data processing with related technologies likes Apache Spark, Apache Storm, Hadoop….

2. Prerequisite

You will need to install Java 7 or newer versions first.

3. Download and Install Apache Kafka

3.1. Obtain a desire version of Apache Kafka at its site

Download Apache Kafka

You can download a binary version of Apache Kafka by using a web browser or get the mirror link and download with command line tools such as wget, curl, etc. For example, the below *wget* command will download the Apache Kafka 1.0.0 and put in the ***/tmp*** folder:

```
wget http://www.trieuvan.com/apache/kafka/1.0.0/kafka_2.11-1.0.0.tgz -P /tmp
```

3.2. Extract the downloaded file

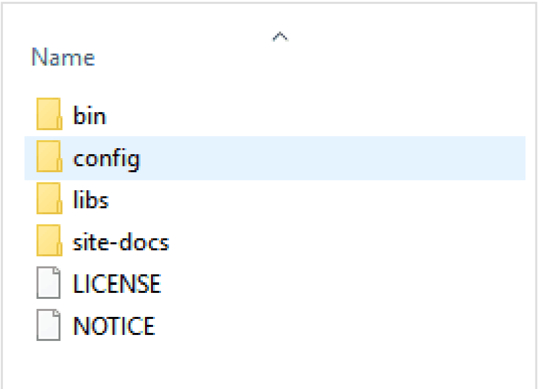
Let's extract the downloaded file to the ***/opt*** directory:

```
sudo tar -xvzf /tmp/kafka_2.11-1.0.0.tgz  -C /opt/
```

3.3. Go to the extracted Kafka directory

```
cd /opt/kafka_2.11-1.0.0/
```

Let's take a look at Kafka directory:



Kafka directory

- bin

  Contains all necessary scripts for working with Apache Kafka such as: start, stop ZooKeeper, Kafka server, create Apache Kafka producer, consumer..

- libs

  Contains all libraries used by Kafka

- config

  Contains all configuration files for ZooKeeper, Logger, Kafka server…

- **site-docs**

  Apache Kafka documents.

4. Start Apache Kafka server

Apache Kafka can be deployed into 3 types of cluster.

- Single node – Single broker cluster

  For this type, we can deploy only a single Kafka broker on a single node.

- Single node – Multiple broker clusters

  This type supports multiple brokers on a Single node. We just need to start Kafka brokers on different ports.

- Multiple nodes – Multiple broker clusters

  This type is combined from above types. We can deploy multiple brokers on each node and connect those nodes together.

In tutorial, we will use the first type, deploy a Single Kafka broker on Single node which is our PC.

Start Apache Kafka requires 2 steps:

- Start ZooKeeper

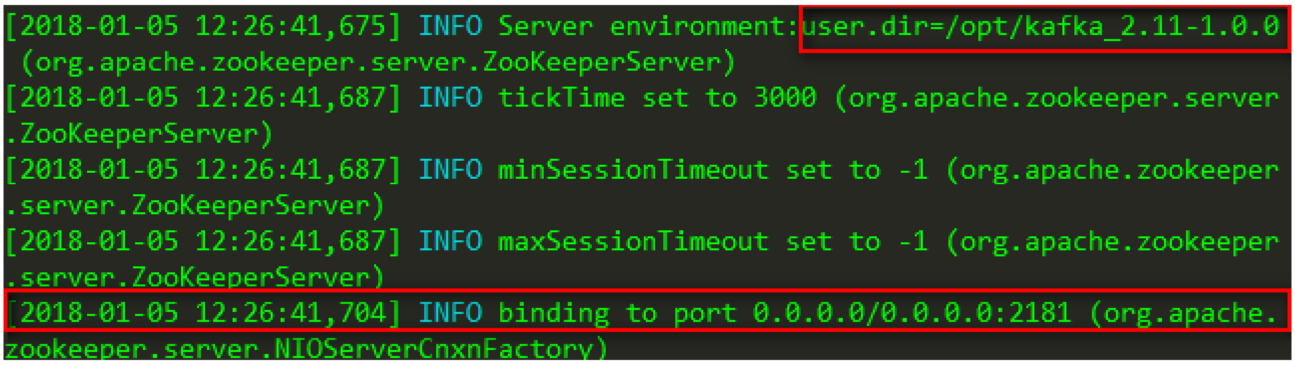
Apache Kafka utilizes ZooKeeper to store metadata of cluster and consumer information. So, we have 2 choices: deploy a separated ZooKeeper server and configure Kafka to use that server. Or simply use the ZooKeeper bundled with Kakfa distribution. In this article, we will chose the option 2, using the ZooKeeper bundled with Kafka

- Start Kafka Broker

4.1. Start ZooKeeper

Let's start the ZooKeeper server by issuing the following command:

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

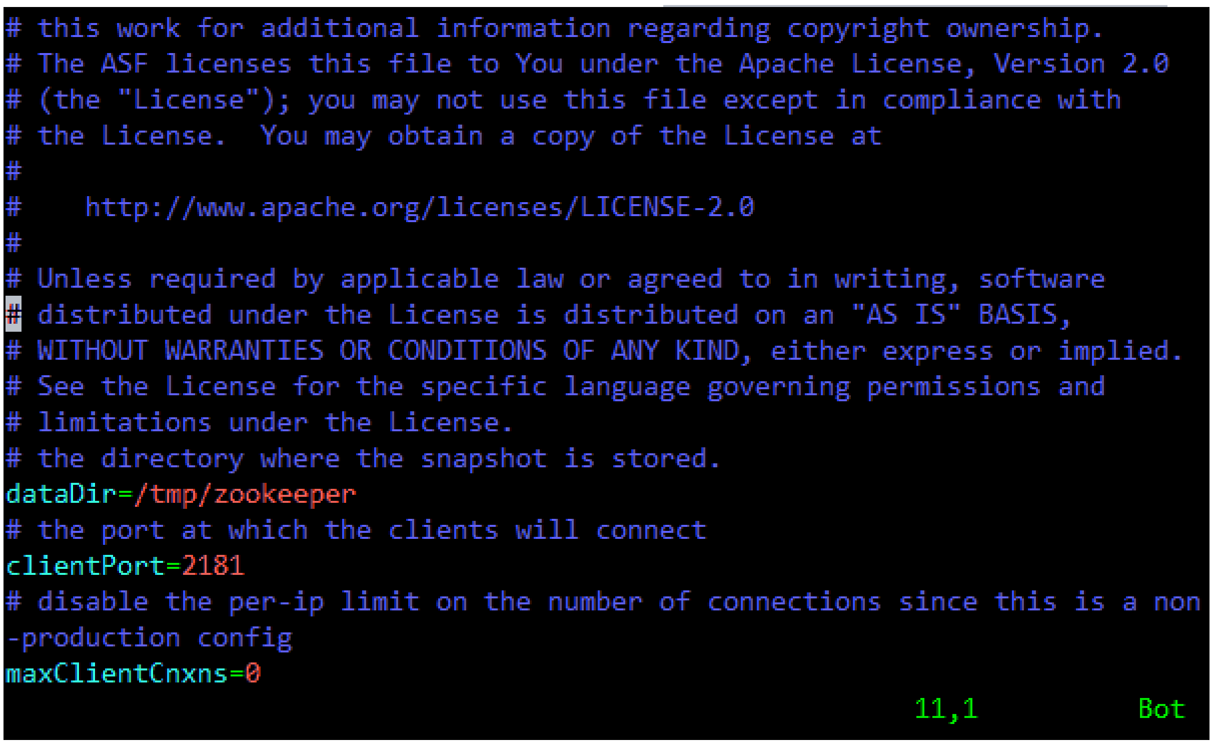


```
[2018-01-05 12:26:41,675] INFO Server environment:user.dir=/opt/kafka_2.11-1.0.0
 (org.apache.zookeeper.server.ZooKeeperServer)
[2018-01-05 12:26:41,687] INFO tickTime set to 3000 (org.apache.zookeeper.server
.ZooKeeperServer)
[2018-01-05 12:26:41,687] INFO minSessionTimeout set to -1 (org.apache.zookeeper
.server.ZooKeeperServer)
[2018-01-05 12:26:41,687] INFO maxSessionTimeout set to -1 (org.apache.zookeeper
.server.ZooKeeperServer)
[2018-01-05 12:26:41,704] INFO binding to port 0.0.0.0/0.0.0.0:2181 (org.apache.
zookeeper.server.NIOServerCnxnFactory)
```

Start ZooKeeper Successfully.

Successful start is ended with info likes: *binding port 0.0.0.0/0.0.0.0:2181*

By default, ZooKeeper listens on the port *2181*. We can change it to another one by editing the file *config/zookeeper.properties*. Let's take a look at its content.



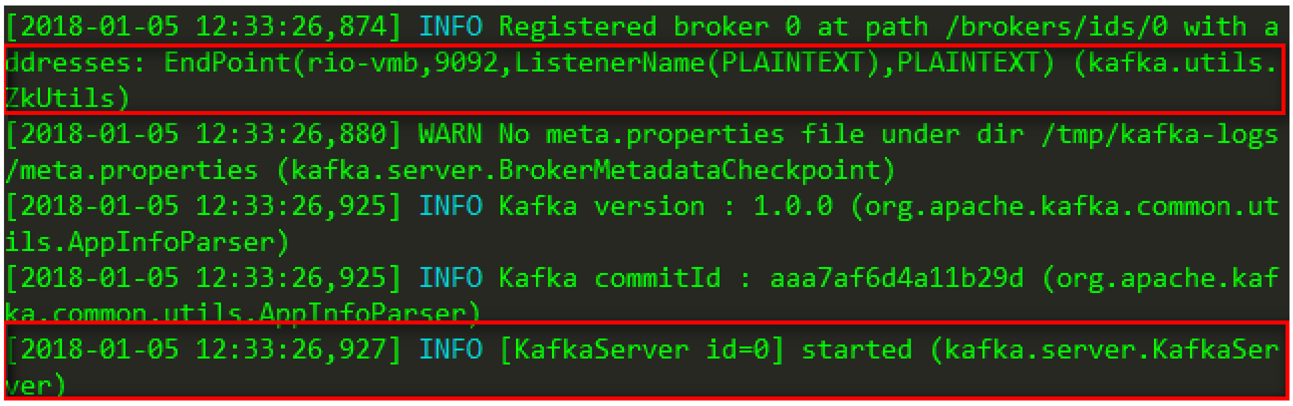
```
# this work for additional information regarding copyright ownership.
# The ASF licenses this file to You under the Apache License, Version 2.0
# (the "License"); you may not use this file except in compliance with
# the License.  You may obtain a copy of the License at
#
#    http://www.apache.org/licenses/LICENSE-2.0
#
# Unless required by applicable law or agreed to in writing, software
# distributed under the License is distributed on an "AS IS" BASIS,
# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
# See the License for the specific language governing permissions and
# limitations under the License.
# the directory where the snapshot is stored.
dataDir=/tmp/zookeeper
# the port at which the clients will connect
clientPort=2181
# disable the per-ip limit on the number of connections since this is a non
-production config
maxClientCnxns=0
                                                                11,1          Bot
```

zookeeper.properties

4.2. Start Kafka Broker

To start Kafka broker, we can issue the below command:

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



```
[2018-01-05 12:33:26,874] INFO Registered broker 0 at path /brokers/ids/0 with a
ddresses: EndPoint(rio-vmb,9092,ListenerName(PLAINTEXT),PLAINTEXT) (kafka.utils.
ZkUtils)
[2018-01-05 12:33:26,880] WARN No meta.properties file under dir /tmp/kafka-logs
/meta.properties (kafka.server.BrokerMetadataCheckpoint)
[2018-01-05 12:33:26,925] INFO Kafka version : 1.0.0 (org.apache.kafka.common.ut
ils.AppInfoParser)
[2018-01-05 12:33:26,925] INFO Kafka commitId : aaa7af6d4a11b29d (org.apache.kaf
ka.common.utils.AppInfoParser)
[2018-01-05 12:33:26,927] INFO [KafkaServer id=0] started (kafka.server.KafkaSer
ver)
```

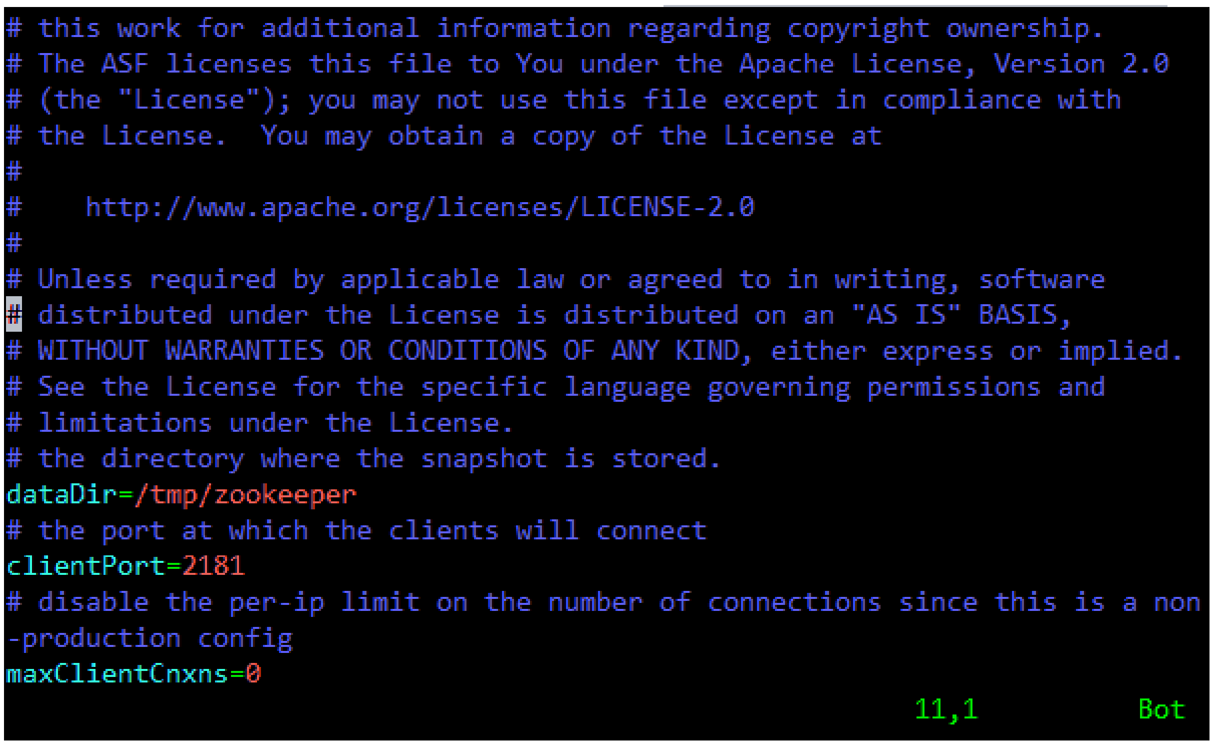
Start Kafka Broker Successfully.

Successful start will ended with some info likes:

```
Registered broker 0 at path/brokers/ids/0 with address: PLAINTEXT --> EndPoint(rio-vmb,9092,PLAINTEXT)
```

Where *rio-vmb* is the hostname of the current server, 9092 is the default port on which Apache Kafka is listened on and the default protocal is PLAINTEXT.

We can change above info by editing the *config/server.properties*. Let's take a look at its content:



```
# this work for additional information regarding copyright ownership.
# The ASF licenses this file to You under the Apache License, Version 2.0
# (the "License"); you may not use this file except in compliance with
# the License.  You may obtain a copy of the License at
#
#    http://www.apache.org/licenses/LICENSE-2.0
#
# Unless required by applicable law or agreed to in writing, software
# distributed under the License is distributed on an "AS IS" BASIS,
# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
# See the License for the specific language governing permissions and
# limitations under the License.
# the directory where the snapshot is stored.
dataDir=/tmp/zookeeper
# the port at which the clients will connect
clientPort=2181
# disable the per-ip limit on the number of connections since this is a non
-production config
maxClientCnxns=0
                                                        11,1          Bot
```

Kafka broker configuration file – server.properties

Some info we can see on the top of the file are: dataDir, clientPort,..

5. Working with Apache Kafka

5.1. Create a Kafka Topic

Let’s create an topic called HelloKafkaTopic by issuing the following command:

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

The output on the console should be:

```
Created topic "HelloKafkaTopic".
```

5.2. To verify that the topic was created, we can list all the topics on the Kafka

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

The output for this command should be:

```
HelloKafkaTopic
```

5.3. Start a Kafka Producer

By default, *Apache Kaka* provides us some client tools (scripts) to start producer and send message to the broker. We will use the */bin/kafka-console-producer.sh* to connect to the the topic *HelloKafkaTopic* of the broker *localhost:9092*:

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

Note that after the above command, the console will be waiting to input message from users. We can type any message and hit enter will push the message to the topic.

To exit the producer client, simply press: *ctrl-c*.

5.4. Start a Kafka Consumer to consume the messages produced by above producer

In similar to the producer, we will use the default consumer client (script) provided by Kafka.

```
bin/kafka-console-consumer.sh --zookeeper localhost:2181 --topic HelloKafkaTopic --from-beginning
```

5.5. Produce and Consume message

Back to the Producer console which is still waiting for input from keyboard, and put any message and hit enter. For example,

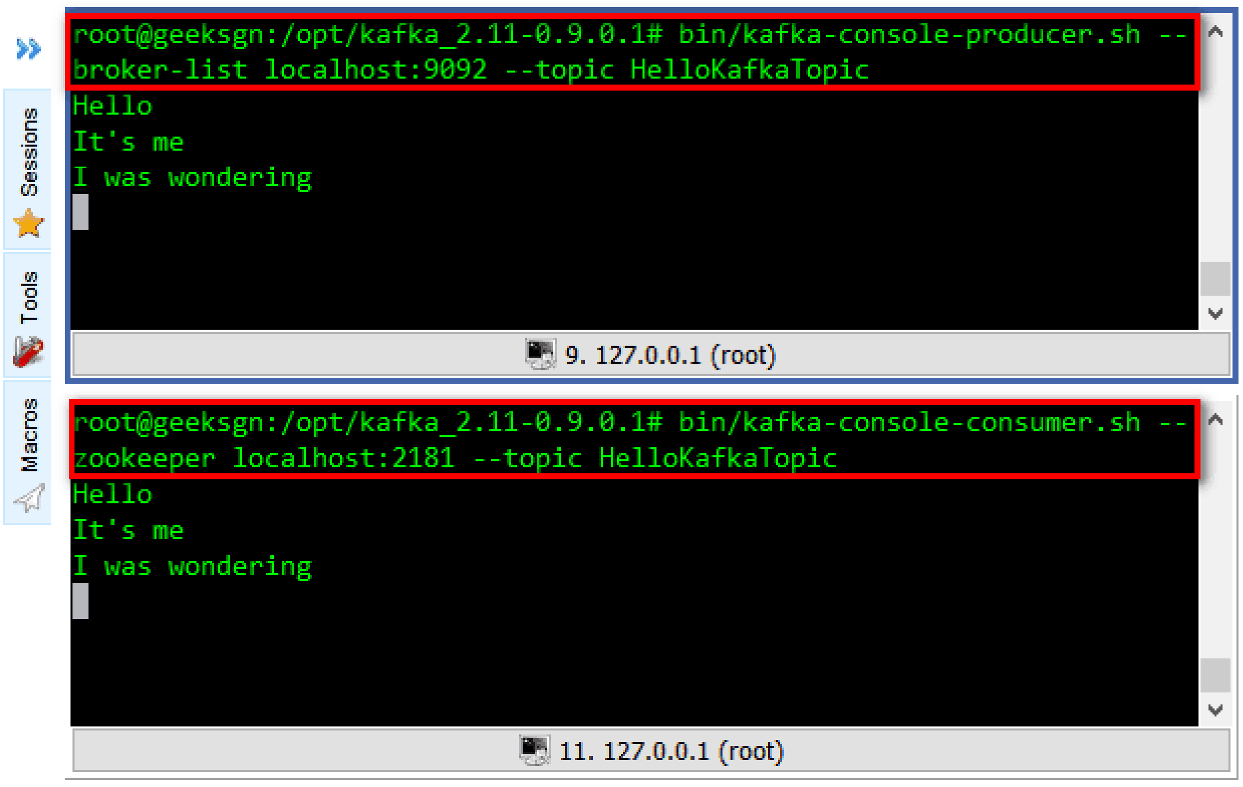
```
Hello

It's me
```

```
I was wondering
```

You can see all those messages are consumed and displayed on the Consumer console.

The conversation will be as below images. The top console is the Producer console while the bottom is the Consumer console.



```
root@geeksgn:/opt/kafka_2.11-0.9.0.1# bin/kafka-console-producer.sh --
broker-list localhost:9092 --topic HelloKafkaTopic
Hello
It's me
I was wondering
```

```
root@geeksgn:/opt/kafka_2.11-0.9.0.1# bin/kafka-console-consumer.sh --
zookeeper localhost:2181 --topic HelloKafkaTopic
Hello
It's me
I was wondering
```

Producer vs Consumer consoles

6. Conclusion

That was all basic steps to started with the **Apache Kafka**. On next article, I will share additional configuration for Apache Kafka broker, how to use other client libraries like Java, C#…to produce and consume messages with Kafka cluster.

Below are the articles related to Apache Kafka topic. If you're interested in them, you can refer to the following links:

Apache Kafka Tutorial

Apache Kafka 0.9 Java Client API Example

How To Write A Custom Serializer in Apache Kafka

Write An Apache Kafka Custom Partitioner

Using Apache Kafka Docker

Apache Kafka Command Line Interface

Apache Kafka Connect Example

Apache Kafka Connect MQTT Source Tutorial

Apache Flume Kafka Source And HDFS Sink Tutorial