Kafka Install Instructions

Contents

[1. Download & Install Gradle 2](#_Toc446061021)

[2. Download & Install Kafka 2](#_Toc446061022)

[2.1. Download Kafka 2](#_Toc446061023)

[2.2. Compile/Build Kafka using Gradle 3](#_Toc446061024)

[3. Start the Kafka Server 3](#_Toc446061025)

[3.1. ZooKeeper Server 3](#_Toc446061026)

[3.2. Kafka Server 3](#_Toc446061027)

[4. Test/Validate Kafka 4](#_Toc446061028)

[4.1. Create a topic 4](#_Toc446061029)

[4.2. Send some messages 4](#_Toc446061030)

[4.3. Start a consumer 4](#_Toc446061031)

[5. Links 5](#_Toc446061032)

# Download & Install Gradle

*A pre-requisite to installing Kafka is for ‘Gradle’ to be installed on the VM.*

*Upon downloading Kakfa, the installation needs to be compiled/built locally – which is done using Gradle.*

*So first, Gradle needs to be downloaded & copied over to the localhost.*

1. Within the Virtual Machine, download Gradle. To do so, open a Web browser & go to the site –

gradle.org/gradle-download

Download a copy of the ‘**complete distribution’**

1. Copy the Gradle install files to the localhost. Type –

cd Downloads/

scp –P 2222 gradle-2.12-all-zip root@localhost:/root

1. ssh into the hadoop cluster. Type –

ssh [root@127.0.0.1](mailto:root@127.0.0.1) –p 2222

1. Unzip the downloaded zip file. Type –

unzip gradle-2.12-all.zip

1. Configure Gradle (set up environment variable & add to Unix PATH variable). Type –

GRADLE\_HOME=~/gradle-2.12

export GRADLE\_HOME

export PATH=$PATH:$PATH:$GRADLE\_HOME/bin

1. Validate that Gradle has been installed correctly by typing –

gradle -v

The output shows the Gradle version & also the local environment configuration.

# Download & Install Kafka

Now Gradle has been installed, Kafka can be downloaded & built

## Download Kafka

1. If you haven’t already done so, ssh into the hadoop cluster. Type –

ssh [root@127.0.0.1](mailto:root@127.0.0.1) –p 2222

1. Change directory to the folder /opt. Type –

cd /opt

1. Download the latest version of Kafka from Git & ‘cd’ to the ‘kafka’ folder generated. Type –

git clone <https://github.com/apache/kafka> kafka

4a. Then type –

cd kafka

## Compile/Build Kafka using Gradle

1. Type –

gradle

1. Build a jar & run it. Type –

./gradlew jar

1. Buid the source jar. Type –

./gradlew srcJar

1. Build javadocs & scaladocs. Type –

./gradlew javadoc

./gradlew javadocJar # builds a jar from the javadocs

./gradlew scaladoc

./gradlew scaladocJar # builds a jar from the scaladocs

./gradlew docsJar # builds both javadoc and scaladoc jar

1. Clean the build. Type –

./gradlew clean

1. Build the jar for all scala versions & for all projects

./gradlew jarAll

# Start the Kafka Server

## ZooKeeper Server

Kafka uses ZooKeeper, however this should already be running as part of the Hortonworks distribution.

## Kafka Server

Now start the Kafka server. Within the ‘kafka’ directory, type –

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

Upon pressing enter; you should first see a number of messages, starting with ‘SLFJ:...’. This is fine, this is the logging being established.

The terminal should have outputted a number of messages, the last saying something similar to –

INFO [Group Metadata Manager on Broker 0]: Removed 0 expired offsets in...

What this means is that the server is now running & you will need to open a new terminal session to test/use the Kafka server. As such –

1. Open a new terminal
2. ssh into the hadoop cluster. Type –

ssh [root@127.0.0.1](mailto:root@127.0.0.1) –p 2222

1. ‘cd’ to the kafka directory. Type –

cd /opt/kafka/

# Test/Validate Kafka

## Create a topic

Create a topic named "test" with a single partition and only one replica:

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

We can now see that topic if we run the list topic command:

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

test

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

## Send some messages

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. By default each line will be sent as a separate message.

Run the producer and then type a few messages into the console to send to the server.

> **bin/kafka-console-producer.sh --broker-list localhost:9092 --topic test**

**This is a message**

**This is another message**

## Start a consumer

Kafka also has a command line consumer that will dump out messages to standard output.

> **bin/kafka-console-consumer.sh --zookeeper localhost:2181 --topic test --from-beginning**

This is a message

This is another message

If you have each of the above commands running in a different terminal then you should now be able to type messages into the producer terminal and see them appear in the consumer terminal.

All of the command line tools have additional options; running the command with no arguments will display usage information documenting them in more detail.

Further documentation & testing can be found on the Kafka quick start guide, link - <http://kafka.apache.org/documentation.html#quickstart>

# Links

Kafka Quickstart instructions - <http://kafka.apache.org/documentation.html#quickstart>

Kafka github page - <https://github.com/apache/kafka>