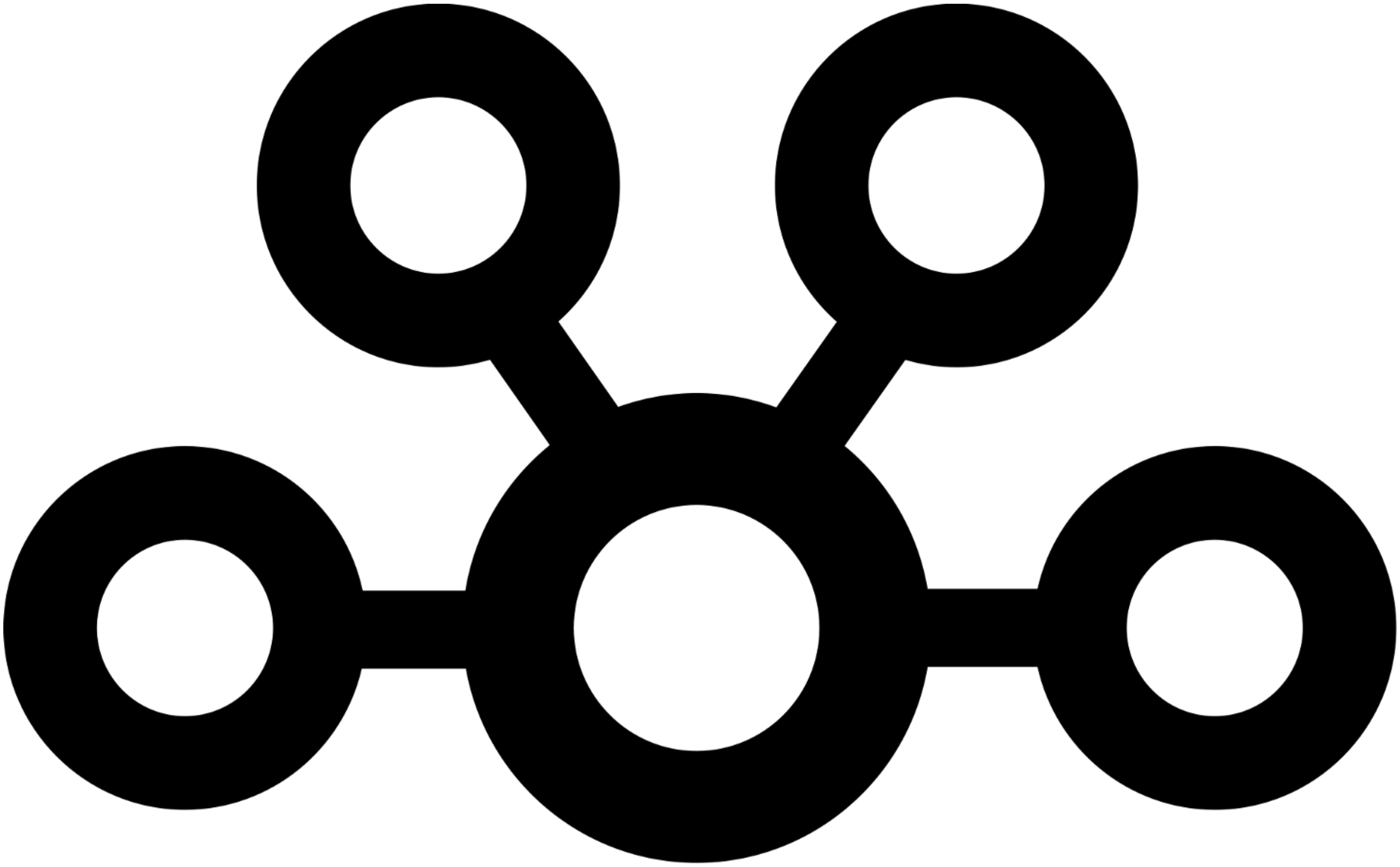

Ejecutando Apache Kafka en Windows 10

Bivás Biswas 19 de noviembre de 2019 · 12 min de lectura ★

> *bancos, ocho de las diez principales compañías de seguros, nueve de las diez principales compañías de telecomunicaciones y mucho más. LinkedIn, Microsoft y Netflix procesan mensajes de cuatro comas al día con Kafka (1.000.000.000.000).*
>
> ***Kafka se utiliza para flujos de datos en tiempo real, para recopilar macrodatos o para realizar análisis en tiempo real (o ambos)*** *. Kafka se utiliza con microservicios en memoria para proporcionar durabilidad y se puede utilizar para alimentar eventos a CEP (sistemas de transmisión de eventos complejos) y sistemas de automatización de estilo IoT / IFTTT. - DZone*

Hay muchos artículos en Internet sobre este tema, pero muchos de ellos están rotos o simplemente se copian y pegan de artículos destinados a Linux que no funcionan en Windows. Aunque es una instalación fácil, hay algunos inconvenientes.

Este artículo lo ayudará a mantenerse alejado de las trampas y a mostrar Kafka en una plataforma Windows 10.

Kafka depende de Zookeeper como su núcleo de mensajería distribuida. Por lo tanto, el servidor del guardián del zoológico debe estar en funcionamiento primero para que el servidor Kafka pueda conectarse a él.

Antes de Dow n de carga Zookeeper y Kafka, asegúrese de que tiene 7-Zip instalado en su sistema. (Gran herramienta para trabajar con archivos tar y gz).

En este artículo, primero, nos aseguraremos de tener algunas herramientas necesarias. Luego instalaremos, configuraremos y ejecutaremos Zookeeper. Después de eso, instalaremos, configuraremos y ejecutaremos Kafka.

Vamos a por ello.

Instalar herramientas

JRE. Luego configure sus variables HOME y PATH. Pero primero, 7-zip.

Instalar 7-zip

Puede encontrar 7-zip aquí -

Descargar

Inglés Chino Simpl. Tradicional chino. Esperanto Francés Alemán Indonesio Japonés Portugués Brasil Español Tailandés…

www.7-zip.org

When you install 7-zip, make sure to add it to your right-click menu shortcuts. It'll make your life easier.

Install JDK

It used to be free, and it still is except now you need to create an account with Oracle to be able to download this. Here's the link-

Java SE Downloads

Java SE 13.0.1 is the latest release for the Java SE Platform Learn more Looking for Oracle OpenJDK builds? Oracle…

www.oracle.com

JDK will first install itself and then it'll ask you for the location where you want to install JRE.

The default install location is "C:\Program Files\…"

Normally it's fine to put Java in there but best not to for your Kafka installation. Kafka was meant for Linux. I wouldn't push it with Windows directory names containing spaces.

We'll need to set PATHs for all of our installations. So we'll be revisiting this a few times in this article. We'll set two kinds of PATH variables — the User variable and the System variable. In the User variable, we'll add the path to the install location, and in the system variable, we'll set the path to the bin directory.

Type 'Environment' in your Windows search bar. It'll come up with an option to Edit the Systems variable in the Control Panel. Click that and you'll be in System Properties. Click on the button that says Environment Variables… on the lower right.

User Variables

Under User variables which is the top box, click on New and add JAVA_HOME and JRE_HOME. For the value for JAVA_HOME, click on the Browse Directories button and navigate to where you installed JDK. I used C:\Java\jdk1.8.0_221\

For the value of your JRE_HOME, click on the Browse Directories button and navigate to where you installed JRE. For me, that is C:\Java\jre1.8.0_221\

System variables

In the System variable box, double click on the Path and add the following two entries at the end

%JAVA_HOME%\bin

%JRE_HOME%\bin

Test

If you've set it up right, open a command prompt in any directory. (A quick way to do that is to go to any directory with your windows file explorer and then type 'cmd' (without the quotes) in the address bar of the file explorer. It'll open up a prompt at that location.)

Navigate to a directory that's different from your Java installation so you really know if your PATH variables are working.

Type java -version

```
C:\>java -version
java version “1.8.0_221”
Java(TM) SE Runtime Environment (build 1.8.0_221-b11)
Java HotSpot(TM) 64-Bit Server VM (build 25.221-b11, mixed mode)
```

If you get something else like *java is not recognized as an internal or external command,* then your PATH is not set correctly. To make sure that it's not a problem with the installation itself, you can go to the bin directory of the JDK and type in the above command. If it works from there but not from anywhere else, it means the system can't find it.

For every command typed in the command prompt, your computer runs through the list in your PATH variables to find a match. That's why errors like this are usually PATH problems.

Install Zookeeper

Download the Zookeeper binary. Here is one of the mirrors to download from

Make sure to download the file that has bin in the name and not the other one. If you download the non-bin one then you'll get an error when trying to start the server.

```
http://apache-mirror.8birdsvideo.com/zookeeper/stable/
```

Right-click on the file, and extract it at the same location using 7-zip. This will extract the tar file which is still not the real deal so it doesn't matter where you extract this file.

For the next step, the location will matter:

Right-click on the tar file and extract it to a directory that doesn't have spaces in the name. I put mine in C:\Apache\
So I end up with a directory like. so-

As of this writing, the stable version of zookeeper is 3.5.6. Yours may be different.

Note the -bin appended to the name. If you don't see this then you've downloaded and extracted the wrong file. Go back to the mirror.
This is important otherwise when you start the Zookeeper server, you'll get an error that looks like this

```
Error: Could not find or load main class
org.apache.zookeeper.server.quorum.QuorumPeerMain
```

Configure Zookeeper

All the configurations happen in just one file- the configuration file, and it's in the conf directory.

Go to the conf directory of your zookeeper install. For me, it's at

```
C:\Apache\apache-zookeeper-3.5.6-bin\conf
```

Rename the zoo_sample.cfg file to zoo.cfg

Open it with a text editor.

In this file, you'll see an entry called the dataDir with a value of /tmp. It pretty much tells you what to do

```
# the directory where the snapshot is stored.
# do not use /tmp for storage, /tmp here is just
# example sakes.
```

Most all the articles on the net tell you to replace this line with something like dataDir=:\zookeeper-3.5.6\data

```
ERROR— Unable to access datadir, exiting abnormally
Unable to create data directory :zookeeper-3.5.6data\version-2
Unable to access datadir, exiting abnormally
```

To avoid this error, point your logs to a path which is one level up from the bin directory, like so-

```
dataDir=../logs
```

(it could be any level up or in the same directory. You can also type in an absolute windows path starting from root C:\\)

This will create a logs directory in your zookeeper install directory that'll store the snapshots when you run the server.

The configuration is done. Let's set up PATH variables so the system can find it from anywhere.

Set PATHs

As before, start typing Environment in your Windows search bar. It'll come up with an option to Edit the Systems variable in the Control Panel. Click on that and you'll be in System Properties. Click on the button that says Environment Variables… on the lower right.

User Variables

Under User variables which is the top box, click on New and add ZOOKEEPER_HOME. For the value, click on the Browse Directories button and navigate to where you installed Zookeeper. For me, that is C:\Apache\apache-zookeeper-3.5.6-bin

System variables

In the System variable box, double click on the Path and add the following at the end

%ZOOKEEPER_HOME%\bin

zkserver

It'll start spewing out a whole bunch of messages. Some of the interesting ones to note are below. (I've cleaned up the verbiage for visual clarity). They look like this

2019–11–19 11:17:17,986 [myid:] — INFO -> *for information*

2019–11–19 11:17:17,986 [myid:] — WARN -> *for warnings*

> *myid is empty because I don't have a myid file in my dataDir. Zookeeper keeps track of every machine in a cluster by their id. To assign an id to a machine, simply place a file name myid (without any extensions) that contains a single number. I'm running Zookeeper in a single server mode for development so setting an id is not necessary. But, if I create a file with the number 5 (arbitrary but needs to be unique if you have more than one machine in a cluster) then the command line would look like this-*
>
> *2019–11–19 12:05:21,400* ***[myid:5]*** *— INFO [main:FileSnap@83] — Reading snapshot ..\logs\version-2\snapshot.a6*

```
Server environment:os.name=Windows 10
Server environment:os.arch=amd64
Server environment:os.version=10.0
Server environment:user.name=Bivas Biswas
Server environment:user.home=C:\Users\Bivas Biswas
Server environment:user.dir=C:\Apache\apache-zookeeper-3.5.6-bin\bin
Server environment:os.memory.free=946MB
Server environment:os.memory.max=14491MB
Server environment:os.memory.total=977MB
minSessionTimeout set to 4000
maxSessionTimeout set to 40000
Created server with tickTime 2000 minSessionTimeout 4000
maxSessionTimeout 40000 datadir ..\logs\version-2 snapdir
..\logs\version-2
Logging initialized @5029ms to org.eclipse.jetty.util.log.Slf4jL
```

The reference to log4j in that last line is a reference to the logging infrastructure that zookeeper uses. You'll also notice that it's logging snapshots to the logs directory that we

```
Snapshotting: 0x0 to C:Apachezookeeper-3.5.6-binlogs\version-
2\snapshot.0
```

After a few seconds of spewing data, it should come to these golden lines

```
Started AdminServer on address 0.0.0.0, port 8080 and command URL
/commands
Using org.apache.zookeeper.server.NIOServerCnxnFactory as server
connection factory
Configuring NIO connection handler with 10s sessionless connection
timeout, 3 selector thread(s), 40 worker threads, and 64 kB direct
buffers.
binding to port 0.0.0.0/0.0.0.0:2181
```

Now Zookeeper server is running on localhost:2181. The AdminServer on port 8080 is a new addition. We can use that port on our browser to monitor zookeeper.

However, you can't go to port 2181 where zookeeper is running. Zookeeper is for Kafka to use as a core kernel of a Kafka Cluster. If you navigate to that port on your browser, which will send some TCP traffic to it that it's not expecting, you'll crash the server. This is what you'll get —

```
Exception causing close of session 0x0: Len error 1195725856
```

So, that's it. Your Zookeeper is up and running on Windows 10 without needing to use a docker composer or a Linux VM.

Next Up — Kafka

Kafka is a message broker. It lets you create topics that you can think of are like chat rooms. You post a message on that topic and people who are subscribed to the topic will

Kafka also comes with 2 more capabilities. One is Stream processing API which basically takes these messages and transforms them to a different value before the recipient gets it. This happens in real-time in real-time data streams.

The other is the Connector API that lets Kafka connect to databases or storage systems to store the data. This data can then be used for further processing by clusters like Hadoop, Map Reduce, etc. This can be happening in addition to the real-time delivery of messages to the consumers.

Kafka is an all in one solution today. Previously, you'd have needed a stream processing framework like Apache Storm to transform the stream but with Kafka's native Stream API, we don't necessarily need Storm as much as we used to. It depends on your use case and the topology that makes sense for you but nice to have options.

Install Kafka

Download Kafka from here -

```
http://kafka.apache.org/downloads.html
```

Grab the Binary downloads. In that section, you might see multiple versions marked Scala x.xxx. If you're using Scala as a client then grab the version that matches your Scala version. I'm using NodeJS as my client so it doesn't matter which one I get. As of this writing, Apache recommends kafka_2.12–2.3.1.tgz so that's the one I got.

Use 7-zip to extract the tgz to a tar file. Then use 7-zip to extract the tar file to a location that doesn't have spaces in its path. I use C:\Apache, so after decompression, my Kafka live here -

```
C:\Apache\kafka_2.12-2.3.1
```

Configure Kafka

We're not going to be setting any environment variables for Kafka. Kafka is the one looking for zookeeper and JDK. Even the producers and the consumers live within the Kafka ecosystem. They are not separate applications that'd be looking to find Kafka on your computer. In a nutshell, no environment vars to mess with.

However, there is the configuration file to set.

Go to your Kafka config directory. For me it's at

```
C:\Apache\kafka_2.12-2.3.1\config
```

There is a sample server.properties file that we can start with.

For one broker we just need to set up this one file. If we need multiple brokers then duplicate this file once for each broker. For example, if you need 2 message brokers then you'll end up with server.b1.properties and server.b2.properties.

In each file, you'll change the following —

- The broker id

```
# The id of the broker. This must be set to a unique integer for each
broker.

broker.id=0
```

If you're just using 1 broker then leave it at 0. Nothing to change. If you have another broker then change the id in the other files so they are unique.

```
# The id of the broker. This must be set to a unique integer for each
broker.
```

- Change the log dirs. I keep mine to an absolute path. You can use any Windows-style pathing here

```
# A comma separated list of directories under which to store log
files

log.dirs=C:\Apache\kafka_2.12-2.3.1\logs
```

- The replication factor is like RAID on a hard drive. It replicates the data from one broker into another broker for redundancy and fault-tolerance. For development, I'm going to keep this at 1

Browse through the fields in this file. You'll notice the timeout values, partition values, and default Zookeeper port number which all would come in handy later for debugging if problems arise.

By default, Apache Kafka will run on port **9092** and Apache Zookeeper will run on port **2181.**

With that our configuration for Kafka is done. Let's fire up the server.

Running Kafka

Make sure that Zookeeper server is still running.

Navigate to the bin directory in your Kafka install directory. There you'll see a windows directory, go in there. That's where all the awesome windows utilities are stored. For me, it's here -

Fire up a new terminal window

```
C:\Apache\kafka_2.12-2.3.1\bin\windows
```

running the batch file)

```
kafka-server-start.bat C:\Apache\kafka_2.12-
2.3.1\config\server.properties
```

You'll see some results like these -

```
Client environment:java.compiler=<NA>
(org.apache.zookeeper.ZooKeeper)
Windows 10 (org.apache.zookeeper.ZooKeeper)
Client environment:os.arch=amd64 (org.apache.zookeeper.ZooKeeper)
Client environment:os.version=10.0 (org.apache.zookeeper.ZooKeeper)
Client environment:user.name=Bivas Biswas
Client environment:user.home=C:\Users\Bivas Biswas

Client environment:user.dir=C:\Apache\kafka_2.12-2.3.1\bin\windows
(org.apache.zookeeper.ZooKeeper)
Initiating client connection, connectString=localhost:2181
sessionTimeout=6000
watcher=kafka.zookeeper.ZooKeeperClient$ZooKeeperClientWatcher$@57c75
8ac (org.apache.zookeeper.ZooKeeper)
[ZooKeeperClient Kafka server] Waiting until connected.
(kafka.zookeeper.ZooKeeperClient)
```

At this point if the server times out waiting for Zookeeper to respond, go to the command terminal where you're running Zookeeper and hit enter. Sometimes if Zookeeper is idle for a while, I've seen Kafka timing out.

If all goes well, you'll see some metadata dump from Group Coordinator and offset messages that look like this with a blinking waiting cursor

```
[GroupCoordinator 0]: Preparing to rebalance group console-consumer-
83701 in state PreparingRebalance with old generation 1
(__consumer_offsets-10) (reason: removing member consumer-1-9bf4ef2d-
97af-4e59-964e-5bb57b457289 on heartbeat expiration)
(kafka.coordinator.group.GroupCoordinator)
```

You'll see some activity in your Zookeeper terminal. It might take a new snapshot and start a new log file. At this point your Kafka is up and running.

Wrapping up

Couple things to keep in mind with Zookeeper —

The dataDir which we set up as the logs directory will start to fill up pretty fast with the snapshots.

In my tests, running Kafka for less than 15 minutes with one topic produced two 65MB snapshot files. These snapshots are transactional log files and they get written to every time a change in the node is detected.

Zookeeper will create duplicates at times when it combines a few log files into one larger file but it doesn't clean up the old files. So clean this directory yourself. You can use zkTxnLogToolkit that's in the bin directory to configure log retention policy.

If you're deploying on an EC2 instance on AWS, and you go with the free tier of t2.micro, the server won't start.

This is because the default heap size of zookeeper and Kafka comes to about 1GB and the memory on a t2.micro is 1 GB so it'll complain about insufficient memory space.

Para evitar este error, ejecute Kafka en una instancia t3.medium con 4 GB de memoria en lugar de reducir el tamaño del montón en el archivo de configuración.

No hay nada más frustrante que seguir un artículo cuyos pasos no funcionan como dice. Esa es la razón por la que comencé a escribir esto en primer lugar mientras revisaba toneladas de artículos rotos y arreglaba los errores.

Si recibe algún error al seguir estos pasos, hágamelo saber. Deje un comentario a continuación si tiene problemas y me pondré en contacto con usted.

¡Feliz transmisión!

Por Towards Data Science

Ejemplos prácticos del mundo real, investigaciones, tutoriales y técnicas de vanguardia entregadas de lunes a jueves. Haga del aprendizaje su ritual diario. Echar un vistazo

Reciba este boletín

Los correos electrónicos se enviarán a javimartinalonso@gmail.com .
¿No tú?

DevOps Kafka apache Desarrollo de software Ciencia de los datos