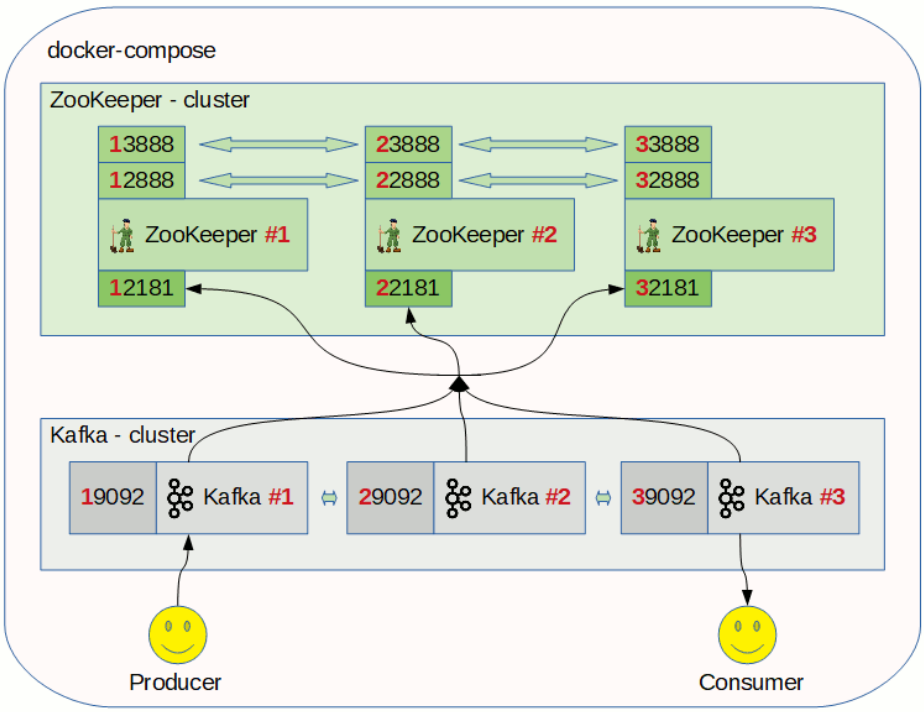
Building Apache Kafka cluster using docker-compose and VirtualBox

lukasz.ciesla (https://better-coding.com/author/lukasz-ciesla/) - June 11, 2018 -
5 Comments (https://better-coding.com/building-apache-kafka-cluster-using-docker-compose-and-virtualbox/#comments)



Apache Kafka is a an open-source stream-processing software platform, designed for high-throughput, low-latency and real-time data broadcasting. It's provided by an easy-scalable and high-availability environment. Let's see how to configure your own docker-compose recipe with fully functional Apache Kafka clustered environment just in few minutes.

Overview

- Preparing host machine for clustered environment using VirtualBox, docker and docker-compose.
- Creating docker-compose recipe file – step by step guide.
- The final version of Apache Kafka cluster docker-compose.yml file.
- Testing Apache Kafka cluster using kafkacat tool.

1. Preparing host machine for clustered environment using VirtualBox, docker and docker-compose

ABOUT ME

ŁUKASZ CIEŚLA

Software Designer, Blogger, Traveller, Photographer

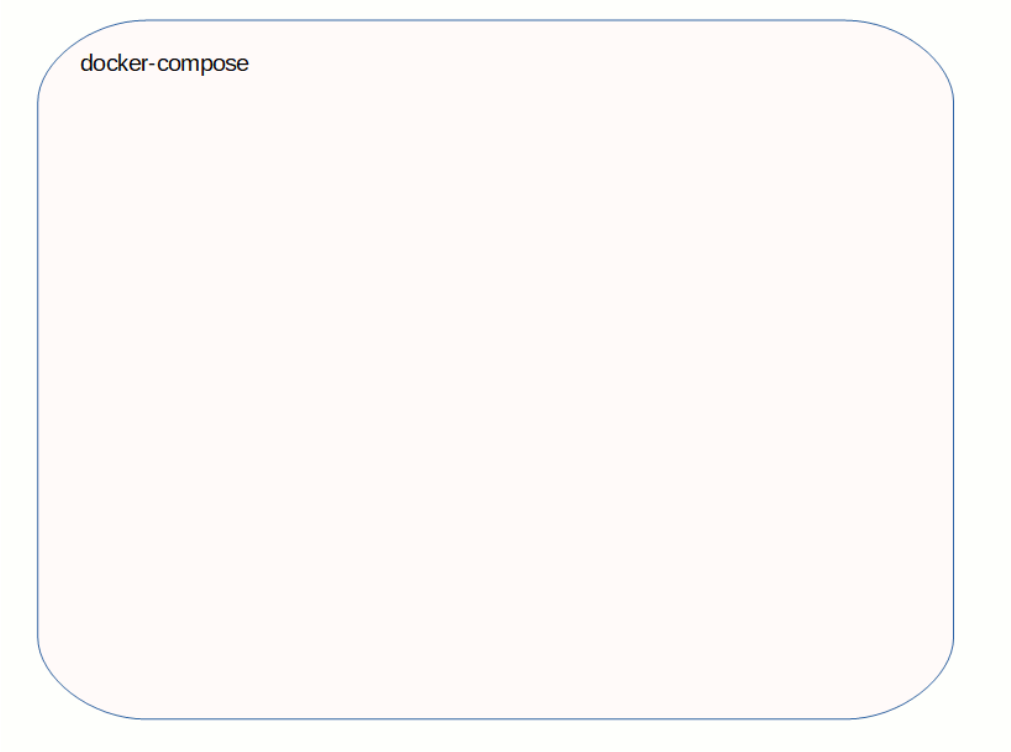
For the purposes of this tutorial I chose Ubuntu Server 18.04 LTS, because it is easy to manage Debian-based Linux distribution with quite fresh versions of tools in its repository.

1.1 Preparing VirtualBox machine

Download ubuntu server and prepare a virtual machine with minimum 10GB hard drive and al least 2GB of RAM. You may follow this tutorial (https://better-coding.com/01-how-to-install-ubuntu-server-on-oracle-virtualbox-in-20-minutes/) if you don't know how to do this.

1.2 Installing docker and docker-compose tools

The next step is to install some necessary tools like docker and docker-compose which allow you to manage many different containers with complex dependencies in a very understandable and compact form.



(https://better-coding.com/wp-content/uploads/2018/06/kafka-cluster-step1.gif)

There are two ways of installing docker in Ubuntu. The first one is by following the official Docker's step by step guide (https://docs.docker.com/install/linux/docker-ce/ubuntu/). The second method is an installation from the official Ubuntu repository. For the purposes of the article I will show you the second one.

Log in to the Ubuntu Server terminal and run following commands:

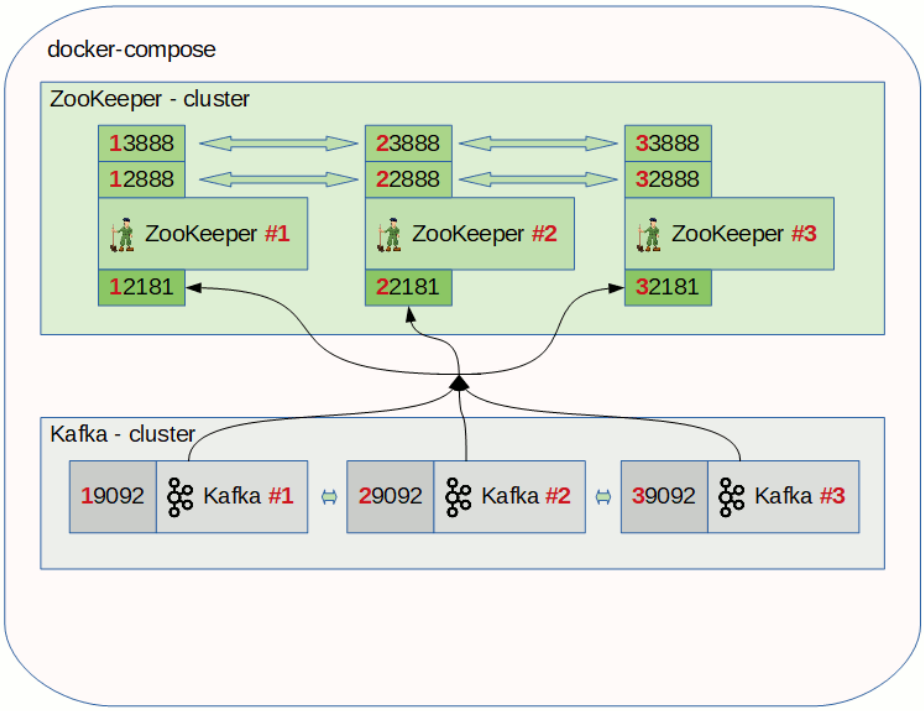
```
sudo apt-get update
sudo apt-get install docker.io docker-compose
```

If you want to use docker command without sudo just add you user to the docker group like it is shown below and restart the virtual machine.

```
sudo usermod -aG docker <user_name>
#Example: sudo usermod -aG docker john
```

2. Creating docker-compose recipe file – step by step guide

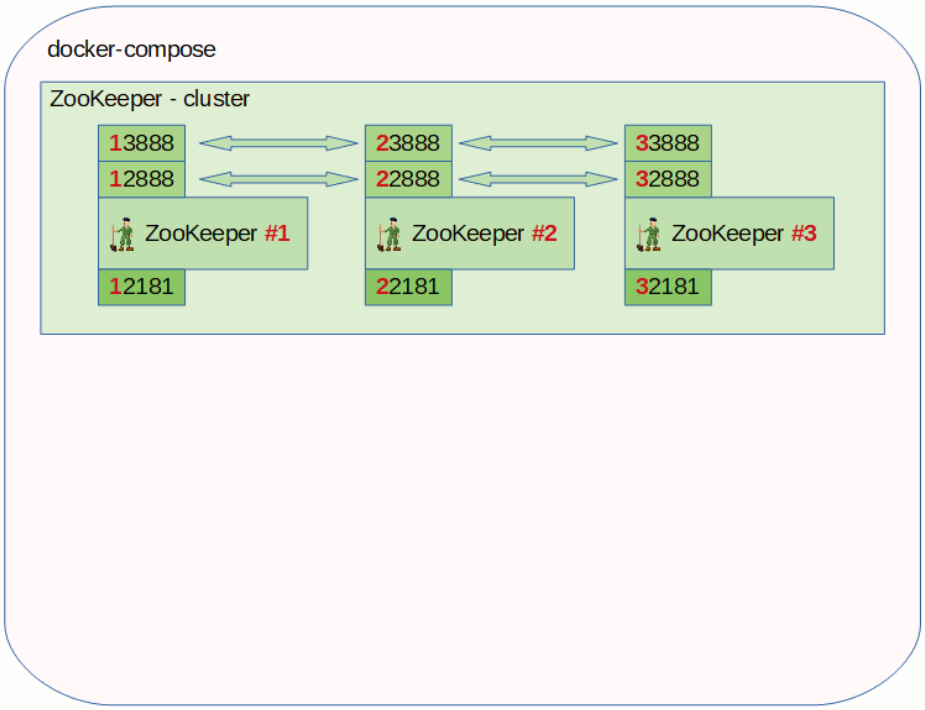
As you can see from the image below, there are two clusters. The first is composed of three ZooKeeper nodes and the second one is consist of three Apache Kafka containers.



(https://better-coding.com/wp-content/uploads/2018/06/kafka-cluster-step2.gif)

2.1 Creating ZooKeepers cluster using docker-compose

Let's look at image below. The ZooKeeper cluster consists of three nodes #1, #2 and #3. Each of them uses two ports (3888 and 2888) for internal cluster communication and exposes 2128 port for clients. Because all of the nodes are located on the same server I have added suffix (node id) to each port to prevent ports collision. The #2888 ports are used for peers communication and the #3888 ports are used for leader elections. More about you can find in the official ZooKeeper documentation (https://zookeeper.apache.org/doc/current/zookeeperStarted.html).



(https://better-coding.com/wp-content/uploads/2018/06/kafka-cluster-step2.1.gif)

It is time to prepare docker-compose recipe file. Let's prepare an empty directory for our Apache Kafka cluster and create docker-compose.yml file with the following content.

DL4J (HTTPS://BETTER-CODING.COM/TAG/DL4J-EN/)
DOCKER (HTTPS://BETTER-CODING.COM/TAG/DOCKER/)
ERROR (HTTPS://BETTER-CODING.COM/TAG/ERROR/)
EXCEPTION (HTTPS://BETTER-CODING.COM/TAG/EXCEPTION/)
FREE (HTTPS://BETTER-CODING.COM/TAG/FREE/)
GIT (HTTPS://BETTER-CODING.COM/TAG/GIT/)
GITLAB (HTTPS://BETTER-CODING.COM/TAG/GITLAB/)
GRADLE (HTTPS://BETTER-CODING.COM/TAG/GRADLE/)
HOWTO (HTTPS://BETTER-CODING.COM/TAG/HOWTO/)
JAVA (HTTPS://BETTER-CODING.COM/TAG/JAVA/)
JAVAFX (HTTPS://BETTER-CODING.COM/TAG/JAVAFX-EN/)
JDEVELOPER (HTTPS://BETTER-CODING.COM/TAG/JDEVELOPER/)
JDK (HTTPS://BETTER-CODING.COM/TAG/JDK/)
JENKINS (HTTPS://BETTER-CODING.COM/TAG/JENKINS/)
KAFKA (HTTPS://BETTER-CODING.COM/TAG/KAFKA/)
LINUX (HTTPS://BETTER-CODING.COM/TAG/LINUX/)
LOAD (HTTPS://BETTER-CODING.COM/TAG/LOAD/)
MACHINELEARNING (HTTPS://BETTER-CODING.COM/TAG/MACHINELEARNING-EN/)
MOOC (HTTPS://BETTER-CODING.COM/TAG/MOOC/)
NETBEANS (HTTPS://BETTER-CODING.COM/TAG/NETBEANS/)
NODEJS (HTTPS://BETTER-CODING.COM/TAG/NODEJS/)
ORACLE (HTTPS://BETTER-CODING.COM/TAG/ORACLE/)
PLSQL (HTTPS://BETTER-CODING.COM/TAG/PLSQL/)
PROGRAMMING (HTTPS://BETTER-CODING.COM/TAG/PROGRAMMING/)
PYTHON (HTTPS://BETTER-CODING.COM/TAG/PYTHON/)
SCM (HTTPS://BETTER-CODING.COM/TAG/SCM/)
SERVLET (HTTPS://BETTER-CODING.COM/TAG/SERVLET/)
SOA (HTTPS://BETTER-CODING.COM/TAG/SOA/)
SOLVED (HTTPS://BETTER-CODING.COM/TAG/SOLVED/)
SPRING (HTTPS://BETTER-CODING.COM/TAG/SPRING/)
SPRINGBOOT (HTTPS://BETTER-CODING.COM/TAG/SPRINGBOOT/)

```
rsion: '2'
rvices:
zookeeper-1:
  image: confluentinc/cp-zookeeper:latest
  hostname: zookeeper-1
  ports:
    - "12181:12181"
  environment:
    ZOOKEEPER_SERVER_ID: 1
    ZOOKEEPER_CLIENT_PORT: 12181
    ZOOKEEPER_TICK_TIME: 2000
    ZOOKEEPER_INIT_LIMIT: 5
    ZOOKEEPER_SYNC_LIMIT: 2
    ZOOKEEPER_SERVERS: zookeeper-1:12888:13888;zookeeper-2:22888:23888;zookeeper-3:328

zookeeper-2:
  image: confluentinc/cp-zookeeper:latest
  hostname: zookeeper-2
  ports:
    - "22181:22181"
  environment:
    ZOOKEEPER_SERVER_ID: 2
    ZOOKEEPER_CLIENT_PORT: 22181
    ZOOKEEPER_TICK_TIME: 2000
    ZOOKEEPER_INIT_LIMIT: 5
    ZOOKEEPER_SYNC_LIMIT: 2
    ZOOKEEPER_SERVERS: zookeeper-1:12888:13888;zookeeper-2:22888:23888;zookeeper-3:328

zookeeper-3:
  image: confluentinc/cp-zookeeper:latest
  hostname: zookeeper-3
  ports:
    - "32181:32181"
  environment:
    ZOOKEEPER_SERVER_ID: 3
    ZOOKEEPER_CLIENT_PORT: 32181
    ZOOKEEPER_TICK_TIME: 2000
    ZOOKEEPER_INIT_LIMIT: 5
    ZOOKEEPER_SYNC_LIMIT: 2
    ZOOKEEPER_SERVERS: zookeeper-1:12888:13888;zookeeper-2:22888:23888;zookeeper-3:328
```

This tutorial bases on Confluent docker-compose.yml (https://github.com/confluentinc/cp-docker-images/blob/master/examples/kafka-cluster/docker-compose.yml) file, but the original Confluent file doesn't allow to connect Kafka from the outside of VirtualBox, because they use dockers host type network. I decided to prepare ready to use version without this issue.

What I have changed:

- I have set hostnames
- I have exposed ports
- I have changed ZOOKEEPER_SERVERS property from localhost to hostnames

2.1.1 Testing if ZooKeeper cluster is running correctly

You can run ZooKeeper cluster by executing the following command:

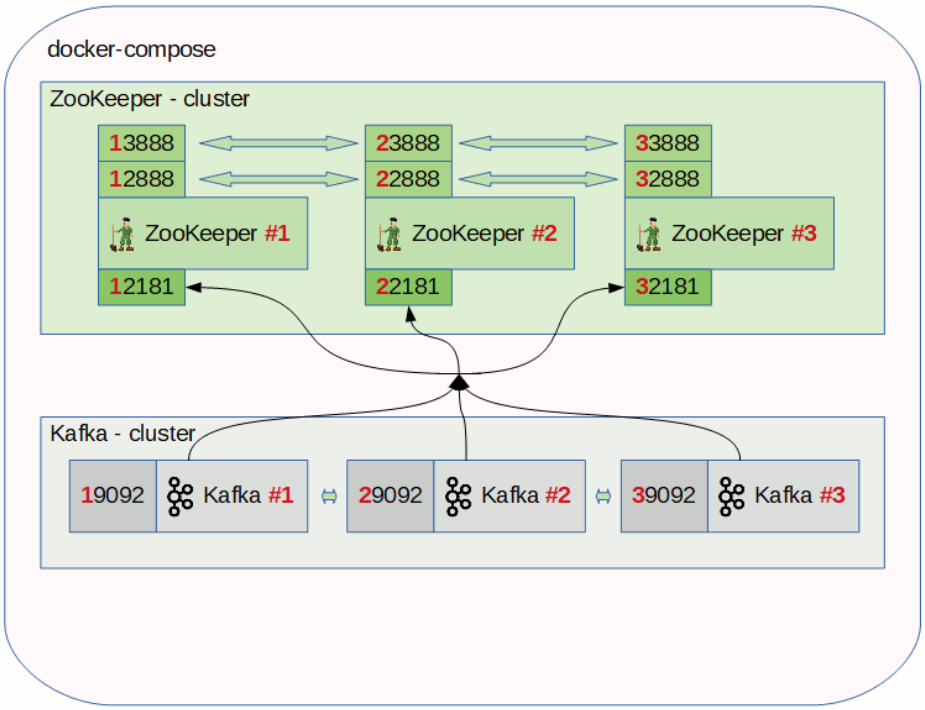
```
docker-compose up
```

And you should see the result like below:

```
zookeeper-2_1  | [2018-06-11 21:11:36,556] INFO Server environment:java.library.path
zookeeper-2_1  | [2018-06-11 21:11:36,556] INFO Server environment:java.io.tmpdir=/t
zookeeper-2_1  | [2018-06-11 21:11:36,556] INFO Server environment:java.compiler=<NA
zookeeper-2_1  | [2018-06-11 21:11:36,556] INFO Server environment:os.name=Linux (or
zookeeper-2_1  | [2018-06-11 21:11:36,556] INFO Server environment:os.arch=amd64 (or
zookeeper-2_1  | [2018-06-11 21:11:36,556] INFO Server environment:os.version=4.15.0
zookeeper-2_1  | [2018-06-11 21:11:36,556] INFO Server environment:user.name=root (r
zookeeper-2_1  | [2018-06-11 21:11:36,557] INFO Server environment:user.home=/root (
zookeeper-2_1  | [2018-06-11 21:11:36,557] INFO Server environment:user.dir=/ (org.a
zookeeper-2_1  | [2018-06-11 21:11:36,559] INFO Created server with tickTime 2000 mi
zookeeper-2_1  | [2018-06-11 21:11:36,560] INFO FOLLOWING - LEADER ELECTION TOOK - 4
zookeeper-2_1  | [2018-06-11 21:11:36,562] INFO Resolved hostname: zookeeper-3 to ad
zookeeper-3_1  | [2018-06-11 21:11:36,565] INFO Follower sid: 2 : info : org.apache.
zookeeper-3_1  | [2018-06-11 21:11:36,590] INFO Synchronizing with Follower sid: 2 m
zookeeper-3_1  | [2018-06-11 21:11:36,590] INFO Sending DIFF (org.apache.zookeeper.s
zookeeper-2_1  | [2018-06-11 21:11:36,591] INFO Getting a diff from the leader 0x100
zookeeper-3_1  | [2018-06-11 21:11:36,595] INFO Received NEWLEADER-ACK message from
zookeeper-3_1  | [2018-06-11 21:11:42,001] INFO Expiring session 0x163f09ed2e90001,
zookeeper-3_1  | [2018-06-11 21:11:42,005] INFO Processed session termination for se
zookeeper-3_1  | [2018-06-11 21:11:42,006] INFO Creating new log file: log.200000001
zookeeper-1_1  | [2018-06-11 21:11:42,006] WARN Got zxid 0x200000001 expected 0x1 (o
zookeeper-2_1  | [2018-06-11 21:11:42,007] WARN Got zxid 0x200000001 expected 0x1 (o
zookeeper-2_1  | [2018-06-11 21:11:42,008] INFO Creating new log file: log.200000001
zookeeper-1_1  | [2018-06-11 21:11:42,010] INFO Creating new log file: log.200000001
```

2.2 Creating Apache Kafka cluster using docker-compose

It is time to add three more containers to docker-compose.yml file which belongs to the Kafka cluster. Newly created servers are pointed to already prepared ZooKeeper cluster as it is shown on the image below.



(https://better-coding.com/wp-content/uploads/2018/06/kafka-cluster-step2.2a.gif)

I use analogical numbering of ports as in the ZooKeeper cluster. Each kafka node exposes #9092 client port.

2.2.1 The final version of Apache Kafka cluster docker-compose.yml file

```
version: '2'
services:
  zookeeper-1:
    image: confluentinc/cp-zookeeper:latest
    hostname: zookeeper-1
    ports:
      - "12181:12181"
    environment:
      ZOOKEEPER_SERVER_ID: 1
      ZOOKEEPER_CLIENT_PORT: 12181
      ZOOKEEPER_TICK_TIME: 2000
      ZOOKEEPER_INIT_LIMIT: 5
      ZOOKEEPER_SYNC_LIMIT: 2
      ZOOKEEPER_SERVERS: zookeeper-1:12888:13888;zookeeper-2:22888:23888;zookeeper-3

  zookeeper-2:
    image: confluentinc/cp-zookeeper:latest
    hostname: zookeeper-2
    ports:
      - "22181:22181"
    environment:
      ZOOKEEPER_SERVER_ID: 2
      ZOOKEEPER_CLIENT_PORT: 22181
      ZOOKEEPER_TICK_TIME: 2000
      ZOOKEEPER_INIT_LIMIT: 5
      ZOOKEEPER_SYNC_LIMIT: 2
      ZOOKEEPER_SERVERS: zookeeper-1:12888:13888;zookeeper-2:22888:23888;zookeeper-3

  zookeeper-3:
    image: confluentinc/cp-zookeeper:latest
    hostname: zookeeper-3
    ports:
      - "32181:32181"
    environment:
      ZOOKEEPER_SERVER_ID: 3
      ZOOKEEPER_CLIENT_PORT: 32181
      ZOOKEEPER_TICK_TIME: 2000
      ZOOKEEPER_INIT_LIMIT: 5
      ZOOKEEPER_SYNC_LIMIT: 2
      ZOOKEEPER_SERVERS: zookeeper-1:12888:13888;zookeeper-2:22888:23888;zookeeper-3

  kafka-1:
    image: confluentinc/cp-kafka:latest
    hostname: kafka-1
    ports:
      - "19092:19092"
    depends_on:
      - zookeeper-1
      - zookeeper-2
      - zookeeper-3
    environment:
      KAFKA_BROKER_ID: 1
      KAFKA_ZOOKEEPER_CONNECT: zookeeper-1:12181,zookeeper-2:12181,zookeeper-3:12181
```

(https://web.whatsapp.com/send?text=Building%20Apache%20Kafka%20cluster%20using%20docker-compose%20and%20VirtualBox https%3A%2F%2Fbetter-coding.com%2Fbuilding-apache-kafka-cluster-using-docker-compose-and-virtualbox%2F)

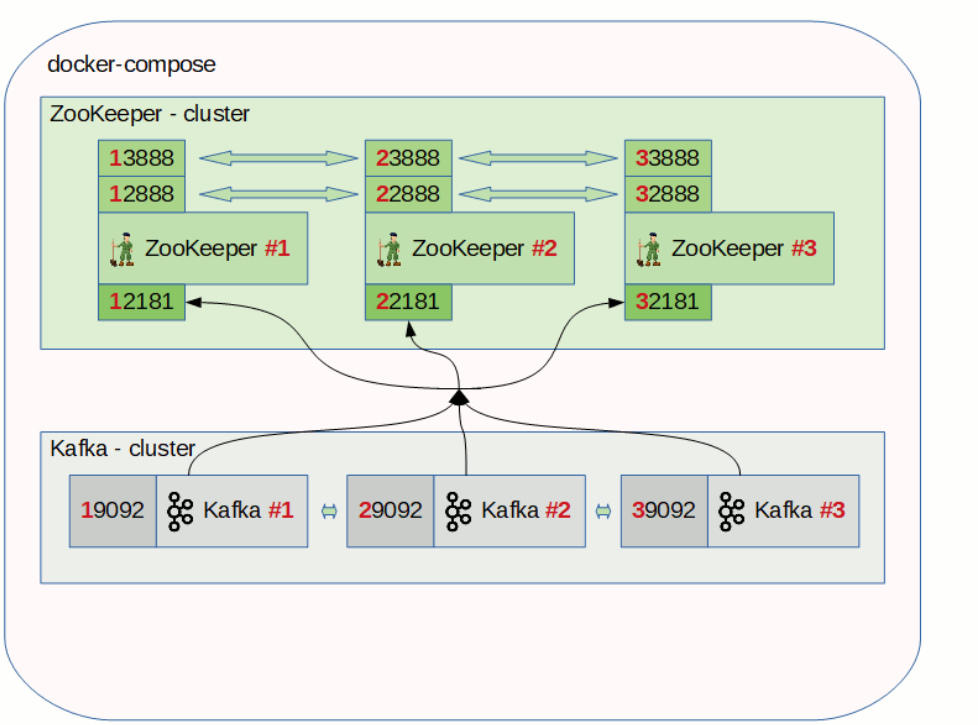
```
    KAFKA_ADVERTISED_LISTENERS: PLAINTEXT://kafka-1:19092

  kafka-2:
    image: confluentinc/cp-kafka:latest
    hostname: kafka-2
    ports:
      - "29092:29092"
    depends_on:
      - zookeeper-1
      - zookeeper-2
      - zookeeper-3
    environment:
      KAFKA_BROKER_ID: 2
      KAFKA_ZOOKEEPER_CONNECT: zookeeper-1:12181,zookeeper-2:12181,zookeeper-3:12181
      KAFKA_ADVERTISED_LISTENERS: PLAINTEXT://kafka-2:29092

  kafka-3:
    image: confluentinc/cp-kafka:latest
    hostname: kafka-3
    ports:
      - "39092:39092"
    depends_on:
      - zookeeper-1
      - zookeeper-2
      - zookeeper-3
    environment:
      KAFKA_BROKER_ID: 3
      KAFKA_ZOOKEEPER_CONNECT: zookeeper-1:12181,zookeeper-2:12181,zookeeper-3:12181
      KAFKA_ADVERTISED_LISTENERS: PLAINTEXT://kafka-3:39092
```

3. Testing Apache Kafka cluster using kafkacat tool

In this step I will show you how to use kafkacat tool to test previously created Kafka cluster. We will send a message to the first node of the cluster and we will see if we will receive the same message from the third node of the cluster as it is shown on image below.



(https://better-coding.com/wp-content/uploads/2018/06/kafka-cluster-step3.gif)

Please remember to add kafka-1, kafka-2 and kafka-3 hosts to the client /etc/hosts file.

```
127.0.0.1       localhost


192.168.1.231 kafka-1 kafka-2 kafka-3

# The following lines are desirable for IPv6 capable hosts
::1     ip6-localhost ip6-loopback
```

Now install kafkacat using the following command:

```
sudo apt-get install kafkacat
```

Run the following command to list all available brokers in the cluster:

```
kafkacat -L -b kafka-1:19092
```


As you can see all of three nodes are accessible:

```
better-coding@bc-vbox:~$ kafkacat -L -b kafka-1:19092
Metadata for all topics (from broker 1: kafka-1:19092/1):
 3 brokers:
  broker 2 at kafka-2:29092
  broker 1 at kafka-1:19092
  broker 3 at kafka-3:39092
 2 topics:
  topic "__confluent.support.metrics" with 1 partitions:
    partition 0, leader 2, replicas: 2,3,1, isrs: 2,3,1
  topic "helloworld.t" with 1 partitions:
    partition 0, leader 1, replicas: 1, isrs: 1
```

Open two instances of terminal and run:

```
kafkacat -P -b kafka-1:19092 -t helloworld_topic
```

```
kafkacat -C -b kafka-3:39092 -t helloworld_topic
```

Then write some message to the first terminal, and you should see the same message in the second terminal.

```
better-coding@bc-vbox:~$ kafkacat -C -b kafka-3:39092 -t helloworld_topic
% Reached end of topic helloworld_topic [0] at offset 0
test_message
% Reached end of topic helloworld_topic [0] at offset 1
```

If you think this post is valuable, please leave me +1 or share it. This action will allow me to reach a wider audience.

Thank you.

TAGS: APACHE (HTTPS://BETTER-CODING.COM/TAG/APACHE/), DOCKER (HTTPS://BETTER-CODING.COM/TAG/DOCKER/), KAFKA (HTTPS://BETTER-CODING.COM/TAG/KAFKA/)

PLEASE SHARE THIS

YOU MIGHT ALSO LIKE

(https://better-coding.com/oracle-adf-docker-how-to-run-oracle-xe-database-in-docker-container/)

Oracle Adf & Docker – How to run Oracle XE Database in Docker container (https://better-coding.com/oracle-adf-

(https://web.whatsapp.com/send?text=Building%20Apache%20Kafka%20cluster%20using%20docker-compose%20and%20VirtualBox https%3A%2F%2Fbetter-coding.com%2Fbuilding-apache-kafka-cluster-using-docker-compose-and-virtualbox%2F)

docker-how-to-run-oracle-xe-database-in-docker-container/)

February 7, 2018

5 Leave a Reply

Join the discussion...

5 0 0 3

Subscribe

newest oldest most voted

sesh

Guest

Can you please add how schema registry is added to the above docker-compose

+ 10 − Reply 1 year ago

olga

Guest

Thank you, this actually worked 🙂 The only thing I had to do was replace 192.168.1.231 with 0.0.0.0

+ 0 − Reply 7 months ago

Talend: Kafka and MongoDB – real-time streaming – Big Data & ETL (https://bigdata-etl.com/talend-kafka-and-mongodb-real-tim

[…] https://better-coding.com/building-apache-kafka-cluster-using-docker-compose-and-virtualbox/ (https://better-coding.com/building-apache-kafka-cluster-using-docker-compose-and-virtualbox/) […]

+ 0 − Reply 5 months ago

Talend: Kafka i MongoDB – strumień danych – Big Data & ETL (https://bigdata-etl.com/pl/talend-kafka-i-mongodb-strumien-dany

[…] https://better-coding.com/building-apache-kafka-cluster-using-docker-compose-and-virtualbox/ (https://better-coding.com/building-apache-kafka-cluster-using-docker-compose-and-virtualbox/) […]

+ 0 − Reply 5 months ago

Raghala

Guest

How to create the topic on start up? Please comment on this

+ 0 − Reply 2 months ago

LINKS

(https://web.whatsapp.com/send?text=Building%20Apache%20Kafka%20cluster%20using%20docker-compose%20and%20VirtualBox https%3A%2F%2Fbetter-coding.com%2Fbuilding-apache-kafka-cluster-using-docker-compose-and-virtualbox%2F)

(https://web.whatsapp.com/send?text=Building%20Apache%20Kafka%20cluster%20using%20docker-compose%20and%20VirtualBox https%3A%2F%2Fbetter-coding.com%2Fbuilding-apache-kafka-cluster-using-docker-compose-and-virtualbox%2F)