DEI
DEPARTAMENTO
DE ENGENHARIA INFORMÁTICA
TÉCNICO LISBOA

Tutorial – Distributed Kafka service

Instituto Superior Técnico
Universidade de Lisboa
Integração Empresarial / Enterprise Integration

The goal of this document is to show how to use the Kafka in a cluster environment.

Consider that all the consumers, producers, and topics management indicated in this tutorial are executed, by the scripts delivered in the Kafka distribution, in a command line, but outside the EC2 AWS instance with your Kafka installation.
For that end, create an additional new EC2 AWS instance with Kafka installed without starting its services. Then you can create as many sessions as you need.

The following contents is presented in this document.

Contents

A. Listing all topics

```
sudo /usr/local/kafka/bin/kafka-topics.sh --list --bootstrap-server <YourIP_or_DNS>:9092

CustomerCountry
EXERCISE3
__consumer_offsets
novo_topico_automatico1
novo_topico_automatico2
novo_topico_automatico3
novotopico
test
```

B. Creating one topic with multiple partitions

```
sudo /usr/local/kafka/bin/kafka-topics.sh --create --bootstrap-server <YourIP_or_DNS>:9092 -replication-
factor 1 --partitions 8 --topic clicks

Created topic "clicks".
```

C. Adding partitions for a specific topic

```
sudo /usr/local/kafka/bin/kafka-topics.sh --alter --bootstrap-server <YourIP_or_DNS>:9092 --partitions 10 --
topic clicks

WARNING: If partitions are increased for a topic that has a key, the partition logic or ordering of the
messages will be affected
Adding partitions succeeded!
```

D. Deleting one topic

```
sudo /usr/local/kafka/bin/kafka-topics.sh --delete --bootstrap-server <YourIP_or_DNS>:9092 --topic clicks

Topic clicks is marked for deletion.
Note: This will have no impact if delete.topic.enable is not set to true.
```

E. Listing the count of partitions for a specific topic

```
/usr/local/kafka/bin/kafka-topics.sh --describe --bootstrap-server <YourIP_or_DNS>:9092 --topic clicks

Topic:clicks    PartitionCount:8        ReplicationFactor:1     Configs:
        Topic: clicks   Partition: 0    Leader: 0       Replicas: 0     Isr: 0
        Topic: clicks   Partition: 1    Leader: 0       Replicas: 0     Isr: 0
        Topic: clicks   Partition: 2    Leader: 0       Replicas: 0     Isr: 0
        Topic: clicks   Partition: 3    Leader: 0       Replicas: 0     Isr: 0
        Topic: clicks   Partition: 4    Leader: 0       Replicas: 0     Isr: 0
        Topic: clicks   Partition: 5    Leader: 0       Replicas: 0     Isr: 0
        Topic: clicks   Partition: 6    Leader: 0       Replicas: 0     Isr: 0
        Topic: clicks   Partition: 7    Leader: 0       Replicas: 0     Isr: 0
```

F. Producing messages to a specific topic with multiple partitions

Remember that:

```
/usr/local/kafka/bin/kafka-console-producer.sh --broker-list <YourIP_or_DNS>:9092 --topic clicks
```

G. Consuming messages from a topic with multiple partitions

Remember that:

```
/usr/local/kafka/bin/kafka-console-consumer.sh --bootstrap-server <YourIP_or_DNS>:9092 --topic clicks --from-
beginning
```

H. Listing the consumer groups

```
sudo /usr/local/kafka/bin/kafka-consumer-groups.sh --bootstrap-server <YourIP_or_DNS>:9092 -list

console-consumer-2817
console-consumer-57329
console-consumer-28356
```

I. Listing the consumer-ids from a specific consumer group

```
sudo /usr/local/kafka/bin/kafka-consumer-groups.sh --bootstrap-server <YourIP_or_DNS>:9092 -describe -group
console-consumer-28356

TOPIC           PARTITION  CURRENT-OFFSET  LOG-END-OFFSET  LAG             CONSUMER-ID                                     HOST            CLIENT-ID
clicks          1          -               0               -               consumer-1-4924ed20-7fc8-46e4-909a-35c71d5a79fc /194.210.61.144 consumer-1
clicks          4          -               0               -               consumer-1-4924ed20-7fc8-46e4-909a-35c71d5a79fc /194.210.61.144 consumer-1
clicks          7          -               0               -               consumer-1-4924ed20-7fc8-46e4-909a-35c71d5a79fc /194.210.61.144 consumer-1
clicks          0          -               1               -               consumer-1-4924ed20-7fc8-46e4-909a-35c71d5a79fc /194.210.61.144 consumer-1
clicks          5          -               0               -               consumer-1-4924ed20-7fc8-46e4-909a-35c71d5a79fc /194.210.61.144 consumer-1
clicks          2          -               1               -               consumer-1-4924ed20-7fc8-46e4-909a-35c71d5a79fc /194.210.61.144 consumer-1
clicks          3          -               1               -               consumer-1-4924ed20-7fc8-46e4-909a-35c71d5a79fc /194.210.61.144 consumer-1
clicks          6          -               0               -               consumer-1-4924ed20-7fc8-46e4-909a-35c71d5a79fc /194.210.61.144 consumer-1
```

J. Defining the group-id in a specific consumer

In command line:

```
sudo /usr/local/kafka/bin/kafka-console-consumer.sh --bootstrap-server <YourIP_or_DNS>:9092 --topic clicks --
from-beginning --group group-id-teste
```

In Java:

```
Define the property:
                                   props.put("group.id", "group-id-teste");

as explained in the section C. of P3-Kafka-ClientsJAVA document.
```

K. Creating two brokers in the same EC2 VM

1. Stop the kafka server if running.
2. Create the two configuration files

```
cp /usr/local/kafka/config/server.properties /usr/local/kafka/config/server-1.properties
cp /usr/local/kafka/config/server.properties /usr/local/kafka/config/server-2.properties
```

3. Edit these new files with the following set of properties (if in the same AWS EC2 instance, otherwise the port and log directory could be the same):

```
config/server-0.properties:
    broker.id=0
    listeners=PLAINTEXT://<YourIP_or_DNS>:9093
    offsets.topic.replication.factor=2
    transaction.state.log.replication.factor=2
    transaction.state.log.min.isr=2
    log.dir=/tmp/kafka-logs-0
```

```
config/server-1.properties:
    broker.id=1
    listeners=PLAINTEXT://<YourIP_or_DNS>:9094
    offsets.topic.replication.factor=2
    transaction.state.log.replication.factor=2
    transaction.state.log.min.isr=2
    log.dir=/tmp/kafka-logs-1
```

4. Start Kafka broker 1 with the command referring to server config file #1

```
     sudo /usr/local/kafka/bin/kafka-server-start.sh -daemon /usr/local/kafka/config/server-0.properties
```

5. Start Kafka broker 2 with the command referring to server config file #2

```
     sudo /usr/local/kafka/bin/kafka-server-start.sh -daemon /usr/local/kafka/config/server-1.properties
```

6. Open the in-bound ports 9093 and 9094 in the AWS EC2 console

7. Create a topic with replication factor=2 and multiple partitions

```
sudo /usr/local/kafka/bin/kafka-topics.sh --create --bootstrap-server <YourIP_or_DNS>:9092 -replication-
factor 2 --partitions 2 --topic testing-cluster

Created topic "testing-cluster".
```

L. Checking that cluster of brokers is launched

Executing the following command:

```
                                        ps -ef |grep java |grep server
```

You should receive something similar with (one PID for each broker using each configuration):

```
root      31619      1  4 08:43 pts/0    00:00:08 java -Xmx1G -Xms1G -server -XX:+UseG1GC -XX:MaxGCPauseMillis=20 -
XX:InitiatingHeapOccupancyPercent=35 -XX:+ExplicitGCInvokesConcurrent -Djava.awt.headless=true -Xloggc:/usr/local/kafka/bin/../logs/kafkaServer-
gc.log -verbose:gc -XX:+PrintGCDetails -XX:+PrintGCDateStamps -XX:+PrintGCTimeStamps -XX:+UseGCLogFileRotation -XX:NumberOfGCLogFiles=10 -
XX:GCLogFileSize=100M -Dcom.sun.management.jmxremote -Dcom.sun.management.jmxremote.authenticate=false -Dcom.sun.management.jmxremote.ssl=false -
Dkafka.logs.dir=/usr/local/kafka/bin/../logs -Dlog4j.configuration=file:/usr/local/kafka/bin/../config/log4j.properties -cp
/usr/local/kafka/bin/../libs/activation-1.1.1.jar:/usr/local/kafka/bin/../libs/aopalliance-repackaged-2.5.0-
b42.jar:/usr/local/kafka/bin/../libs/argparse4j-0.7.0.jar:/usr/local/kafka/bin/../libs/audience-annotations-
0.5.0.jar:/usr/local/kafka/bin/../libs/commons-lang3-
3.5.jar:/usr/local/kafka/bin/../libs/compileScala.mapping:/usr/local/kafka/bin/../libs/compileScala.mapping.asc:/usr/local/kafka/bin/../libs/conn
ect-api-2.1.0.jar:/usr/local/kafka/bin/../libs/connect-basic-auth-extension-2.1.0.jar:/usr/local/kafka/bin/../libs/connect-file-
2.1.0.jar:/usr/local/kafka/bin/../libs/connect-json-2.1.0.jar:/usr/local/kafka/bin/../libs/connect-runtime-
2.1.0.jar:/usr/local/kafka/bin/../libs/connect-transforms-2.1.0.jar:/usr/local/kafka/bin/../libs/guava-20.0.jar:/usr/local/kafka/bin/../libs/hk2-
api-2.5.0-b42.jar:/usr/local/kafka/bin/../libs/hk2-locator-2.5.0-b42.jar:/usr/local/kafka/bin/../libs/hk2-utils-2.5.0-
```

```
b42.jar:/usr/local/kafka/bin/../libs/jackson-annotations-2.9.7.jar:/usr/local/kafka/bin/../libs/jackson-core-
2.9.7.jar:/usr/local/kafka/bin/../libs/jackson-databind-2.9.7.jar:/usr/local/kafka/bin/../libs/jackson-jaxrs-base-
2.9.7.jar:/usr/local/kafka/bin/../libs/jackson-jaxrs-json-provider-2.9.7.jar:/usr/local/kafka/bin/../libs/jackson-module-jaxb-annotations-
2.9.7.jar:/usr/local/kafka/bin/../libs/javassist-3.22.0-CR2.jar:/usr/local/kafka/bin/../libs/javax.annotation-api-
1.2.jar:/usr/local/kafka/bin/../libs/javax.inject-1.jar:/usr/local/kafka/bin/../libs/javax.inject-2.5.0-
b42.jar:/usr/local/kafka/bin/../libs/javax.servlet-api-3.1.0.jar:/usr/local/kafka/bin/../libs/javax.ws.rs-api-
2.1.1.jar:/usr/local/kafka/bin/../libs/javax.ws.rs-api-2.1.jar:/usr/local/kafka/bin/../libs/jaxb-api-
2.3.0.jar:/usr/local/kafka/bin/../libs/jersey-client-2.27.jar:/usr/local/kafka/bin/../libs/jersey-common-
2.27.jar:/usr/local/kafka/bin/../libs/jersey-container-servlet-2.27.jar:/usr/local/kafka/bin/../libs/jersey-container-servlet-core-
2.27.jar:/usr/local/kafka/bin/../libs/jersey-hk2-2.27.jar:/usr/local/kafka/bin/../libs/jersey-media-jaxb-
2.27.jar:/usr/local/kafka/bin/../libs/jersey-server-2.27.jar:/usr/local/kafka/bin/../libs/jetty-client-
9.4.12.v20180830.jar:/usr/local/kafka/bin/../libs/jetty-continuation-9.4.12.v20180830.jar:/usr/local/kafka/bin/../libs/jetty-http-
9.4.12.v20180830.jar:/usr/local/kafka/bin/../libs/jetty-io-9.4.12.v20180830.jar:/usr/local/kafka/bin/../libs/jetty-security-
9.4.12.v20180830.jar:/usr/local/kafka/bin/../libs/jetty server-9.4.12.v20180830.jar:/usr/local/kafka/bin/../libs/jetty-servlet-
9.4.12.v20180830.jar:/usr/local/kafka/bin/../libs/jetty-servlets-9.4.12.v20180830.jar:/usr/local/kafka/bin/../libs/jetty-util-
9.4.12.v20180830.jar:/usr/local/kafka/bin/../libs/jopt-simple-5.0.4.jar:/usr/local/kafka/bin/../libs/kafka_2.12-
2.1.0.jar:/usr/local/kafka/bin/../libs/kafka_2.12-2.1.0-sources.jar:/usr/local/kafka/bin/../libs/kafka-clients-
2.1.0.jar:/usr/local/kafka/bin/../libs/kafka-log4j-appender-2.1.0.jar:/usr/local/kafka/bin/../libs/kafka-streams-
2.1.0.jar:/usr/local/kafka/bin/../libs/kafka-streams-examples-2.1.0.jar:/usr/local/kafka/bin/../libs/kafka-streams-scala_2.12-
2.1.0.jar:/usr/local/kafka/bin/../libs/kafka-streams-test-utils-2.1.0.jar:/usr/local/kafka/bin/../libs/kafka-tools-
2.1.0.jar:/usr/local/kafka/bin/../libs/log4j-1.2.17.jar:/usr/local/kafka/bin/../libs/lz4-java-1.5.0.jar:/usr/local/kafka/bin/../libs/maven-
artifact-3.5.4.jar:/usr/local/kafka/bin/../libs/metrics-core-2.2.0.jar:/usr/local/kafka/bin/../libs/osgi-resource-locator-
1.0.1.jar:/usr/local/kafka/bin/../libs/plexus-utils-3.1.0.jar:/usr/local/kafka/bin/../libs/reflections-
0.9.11.jar:/usr/local/kafka/bin/../libs/rocksdbjni-5.14.2.jar:/usr/local/kafka/bin/../libs/scala-library-
2.12.7.jar:/usr/local/kafka/bin/../libs/scala-logging_2.12-3.9.0.jar:/usr/local/kafka/bin/../libs/scala-reflect-
2.12.7.jar:/usr/local/kafka/bin/../libs/slf4j-api-1.7.25.jar:/usr/local/kafka/bin/../libs/slf4j-log4j12-
1.7.25.jar:/usr/local/kafka/bin/../libs/snappy-java-1.1.7.2.jar:/usr/local/kafka/bin/../libs/validation-api-
1.1.0.Final.jar:/usr/local/kafka/bin/../libs/zkclient-0.10.jar:/usr/local/kafka/bin/../libs/zookeeper-
3.4.13.jar:/usr/local/kafka/bin/../libs/zstd-jni-1.3.5-4.jar kafka.Kafka /usr/local/kafka/config/server-0.properties

root     31959     1  3 08:43 pts/0    00:00:06 java -Xmx1G -Xms1G -server -XX:+UseG1GC -XX:MaxGCPauseMillis=20 -
XX:InitiatingHeapOccupancyPercent=35 -XX:+ExplicitGCInvokesConcurrent -Djava.awt.headless=true -Xloggc:/usr/local/kafka/bin/../logs/kafkaServer-
gc.log -verbose:gc -XX:+PrintGCDetails -XX:+PrintGCDateStamps -XX:+PrintGCTimeStamps -XX:+UseGCLogFileRotation -XX:NumberOfGCLogFiles=10 -
XX:GCLogFileSize=100M -Dcom.sun.management.jmxremote -Dcom.sun.management.jmxremote.authenticate=false -Dcom.sun.management.jmxremote.ssl=false -
Dkafka.logs.dir=/usr/local/kafka/bin/../logs -Dlog4j.configuration=file:/usr/local/kafka/bin/../config/log4j.properties -cp
/usr/local/kafka/bin/../libs/activation-1.1.1.jar:/usr/local/kafka/bin/../libs/aopalliance-repackaged-2.5.0-
b42.jar:/usr/local/kafka/bin/../libs/argparse4j-0.7.0.jar:/usr/local/kafka/bin/../libs/audience-annotations-
0.5.0.jar:/usr/local/kafka/bin/../libs/commons-lang3-
3.5.jar:/usr/local/kafka/bin/../libs/compileScala.mapping:/usr/local/kafka/bin/../libs/compileScala.mapping.asc:/usr/local/kafka/bin/../libs/conn
ect-api-2.1.0.jar:/usr/local/kafka/bin/../libs/connect-basic-auth-extension-2.1.0.jar:/usr/local/kafka/bin/../libs/connect-file-
2.1.0.jar:/usr/local/kafka/bin/../libs/connect-json-2.1.0.jar:/usr/local/kafka/bin/../libs/connect-runtime-
2.1.0.jar:/usr/local/kafka/bin/../libs/connect-transforms-2.1.0.jar:/usr/local/kafka/bin/../libs/guava-20.0.jar:/usr/local/kafka/bin/../libs/hk2-
api-2.5.0-b42.jar:/usr/local/kafka/bin/../libs/hk2-locator-2.5.0-b42.jar:/usr/local/kafka/bin/../libs/hk2-utils-2.5.0-
b42.jar:/usr/local/kafka/bin/../libs/jackson-annotations-2.9.7.jar:/usr/local/kafka/bin/../libs/jackson-core-
2.9.7.jar:/usr/local/kafka/bin/../libs/jackson-databind-2.9.7.jar:/usr/local/kafka/bin/../libs/jackson-jaxrs-base-
2.9.7.jar:/usr/local/kafka/bin/../libs/jackson-jaxrs-json-provider-2.9.7.jar:/usr/local/kafka/bin/../libs/jackson-module-jaxb-annotations-
2.9.7.jar:/usr/local/kafka/bin/../libs/javassist-3.22.0-CR2.jar:/usr/local/kafka/bin/../libs/javax.annotation-api-
1.2.jar:/usr/local/kafka/bin/../libs/javax.inject-1.jar:/usr/local/kafka/bin/../libs/javax.inject-2.5.0-
b42.jar:/usr/local/kafka/bin/../libs/javax.servlet-api-3.1.0.jar:/usr/local/kafka/bin/../libs/javax.ws.rs-api-
2.1.1.jar:/usr/local/kafka/bin/../libs/javax.ws.rs-api-2.1.jar:/usr/local/kafka/bin/../libs/jaxb-api-
2.3.0.jar:/usr/local/kafka/bin/../libs/jersey-client-2.27.jar:/usr/local/kafka/bin/../libs/jersey-common-
2.27.jar:/usr/local/kafka/bin/../libs/jersey-container-servlet-2.27.jar:/usr/local/kafka/bin/../libs/jersey-container-servlet-core-
2.27.jar:/usr/local/kafka/bin/../libs/jersey-hk2-2.27.jar:/usr/local/kafka/bin/../libs/jersey-media-jaxb-
2.27.jar:/usr/local/kafka/bin/../libs/jersey-server-2.27.jar:/usr/local/kafka/bin/../libs/jetty-client-
9.4.12.v20180830.jar:/usr/local/kafka/bin/../libs/jetty-continuation-9.4.12.v20180830.jar:/usr/local/kafka/bin/../libs/jetty-http-
9.4.12.v20180830.jar:/usr/local/kafka/bin/../libs/jetty-io-9.4.12.v20180830.jar:/usr/local/kafka/bin/../libs/jetty-security-
9.4.12.v20180830.jar:/usr/local/kafka/bin/../libs/jetty server-9.4.12.v20180830.jar:/usr/local/kafka/bin/../libs/jetty-servlet-
9.4.12.v20180830.jar:/usr/local/kafka/bin/../libs/jetty-servlets-9.4.12.v20180830.jar:/usr/local/kafka/bin/../libs/jetty-util-
9.4.12.v20180830.jar:/usr/local/kafka/bin/../libs/jopt-simple-5.0.4.jar:/usr/local/kafka/bin/../libs/kafka_2.12-
2.1.0.jar:/usr/local/kafka/bin/../libs/kafka_2.12-2.1.0-sources.jar:/usr/local/kafka/bin/../libs/kafka-clients-
2.1.0.jar:/usr/local/kafka/bin/../libs/kafka-log4j-appender-2.1.0.jar:/usr/local/kafka/bin/../libs/kafka-streams-
2.1.0.jar:/usr/local/kafka/bin/../libs/kafka-streams-examples-2.1.0.jar:/usr/local/kafka/bin/../libs/kafka-streams-scala_2.12-
2.1.0.jar:/usr/local/kafka/bin/../libs/kafka-streams-test-utils-2.1.0.jar:/usr/local/kafka/bin/../libs/kafka-tools-
2.1.0.jar:/usr/local/kafka/bin/../libs/log4j-1.2.17.jar:/usr/local/kafka/bin/../libs/lz4-java-1.5.0.jar:/usr/local/kafka/bin/../libs/maven-
artifact-3.5.4.jar:/usr/local/kafka/bin/../libs/metrics-core-2.2.0.jar:/usr/local/kafka/bin/../libs/osgi-resource-locator-
1.0.1.jar:/usr/local/kafka/bin/../libs/plexus-utils-3.1.0.jar:/usr/local/kafka/bin/../libs/reflections-
0.9.11.jar:/usr/local/kafka/bin/../libs/rocksdbjni-5.14.2.jar:/usr/local/kafka/bin/../libs/scala-library-
2.12.7.jar:/usr/local/kafka/bin/../libs/scala-logging_2.12-3.9.0.jar:/usr/local/kafka/bin/../libs/scala-reflect-
2.12.7.jar:/usr/local/kafka/bin/../libs/slf4j-api-1.7.25.jar:/usr/local/kafka/bin/../libs/slf4j-log4j12-
1.7.25.jar:/usr/local/kafka/bin/../libs/snappy-java-1.1.7.2.jar:/usr/local/kafka/bin/../libs/validation-api-
1.1.0.Final.jar:/usr/local/kafka/bin/../libs/zkclient-0.10.jar:/usr/local/kafka/bin/../libs/zookeeper-
3.4.13.jar:/usr/local/kafka/bin/../libs/zstd-jni-1.3.5-4.jar kafka.Kafka /usr/local/kafka/config/server-1.properties
```

The topic can also be described:

```
sudo /usr/local/kafka/bin/kafka-topics.sh --describe --topic testing-cluster --bootstrap-server <Your_IP_or_DNS>:9092

Topic:testing-cluster   PartitionCount:2        ReplicationFactor:2     Configs:
        Topic: testing-cluster  Partition: 0    Leader: 1       Replicas: 1,0   Isr: 1,0
        Topic: testing-cluster  Partition: 1    Leader: 0       Replicas: 0,1   Isr: 0,1
```

M. Producing messages to a cluster of brokers

Remember that:

```
/usr/local/kafka/bin/kafka-console-producer.sh --broker-list <YourIP_or_DNS>:9093,<YourIP_or_DNS>:9094 --
topic testing-cluster
```

N. Consuming messages from a cluster of brokers

Remember that:

```
/usr/local/kafka/bin/kafka-console-consumer.sh --bootstrap-server <YourIP_or_DNS>:9093,<YourIP_or_DNS>:9094 -
-topic testing-cluster
```

O. Creating N brokers in different EC2 VMs

1. Create multiple EC2 VMs adding the following inbound rules for ports: 2888 and 3888
   Hint: to facilitate the creation of multiple instance you can create an image from one instance. Then, when you launch new instances select your new image. With this approach all the kafka installation procedure is performed only once.

2. Install zookeeper and Kafka in each EC2 VM

3. If previously started, then stop the Zookeeper and Kafka servers in each EC2VM

4. In each EC2 VM change Kafka server configuration file in /usr/local/kafka/config/server.properties accordingly:

```
# the broker ID starting on 0 (this is for broker 2 within a 3 brokers cluster)
broker.id=2

# your EC2 VM name to be accessed from outside
listeners=PLAINTEXT://ec2-54-236-47-54.compute-1.amazonaws.com:9092

# replication factor accordingly with the number of brokers
offsets.topic.replication.factor=3
transaction.state.log.replication.factor=3
transaction.state.log.min.isr=3

#zookeeper connectivity (one per EC2 VM of this cluster)
zookeeper.connect=ec2-54-90-57-82.compute-1.amazonaws.com:2181,ec2-54-173-171-63.compute-
1.amazonaws.com:2181,ec2-54-236-47-54.compute-1.amazonaws.com:2181
```

5. In each EC2 VM change the Zookeeper server configuration file in /usr/local/zookeeper/conf/zoo.cfg accordingly:

```
tickTime=2000
dataDir=/var/lib/zookeeper
clientPort=2181
maxClientCnxns=60
initLimit=10
syncLimit=5
server.1=ec2-54-90-57-82.compute-1.amazonaws.com:2888:3888
server.2=ec2-54-173-171-63.compute-1.amazonaws.com:2888:3888
server.3=ec2-54-236-47-54.compute-1.amazonaws.com:2888:3888
```

6. In each EC2 VM create a file with the ID of the Zookeeper server in /var/lib/zookeeper/myid containing only the ID of each specific Zookeeper server, example:

```
[ec2-user@ip-172-31-50-87 conf]$ cd /var/lib/zookeeper/
[ec2-user@ip-172-31-50-87 zookeeper]$ ls -ltr
total 8
-rw-r--r-- 1 root root   2 Feb  2 14:45 myid
-rw-r--r-- 1 root root   4 Feb  2 16:14 zookeeper_server.pid
drwxr-xr-x 2 root root 273 Feb  2 16:19 version-2
[ec2-user@ip-172-31-50-87 zookeeper]$ more myid
3
```

7. In each EC2 VM start the Kafka and Zookeeper servers.
8. Check the available topics:

```
sudo /usr/local/kafka/bin/kafka-topics.sh --list --bootstrap-server ec2-54-90-57-82.compute-
1.amazonaws.com:9092
```

9. Create a new topic, for example in a 3 broker cluster:

```
sudo /usr/local/kafka/bin/kafka-topics.sh --create --bootstrap-server ec2-54-173-171-63.compute-
1.amazonaws.com:9092 -replication-factor 3 --partitions 6 --topic PURCHASE
```

10. Check how the topic has been created:

```
/usr/local/kafka/bin/kafka-topics.sh --describe --bootstrap-server ec2-54-236-47-54.compute-
1.amazonaws.com:9092 --topic PURCHASE
```

11. Start a new consumer (in a different machine, VM or session) for the new topic using the predefined cluster:

```
/usr/local/kafka/bin/kafka-console-producer.sh --broker-list ec2-54-90-57-82.compute-
1.amazonaws.com:9092,ec2-54-173-171-63.compute-1.amazonaws.com:9092,ec2-54-236-47-54.compute-
1.amazonaws.com:9092 --topic PURCHASE
```

12. Start a new producer (in a different machine, VM or session) for the new topic using the predefined cluster and start sending messaging. Verify if they are consumed correctly:

```
/usr/local/kafka/bin/kafka-console-producer.sh --broker-list ec2-54-90-57-82.compute-
1.amazonaws.com:9092,ec2-54-173-171-63.compute-1.amazonaws.com:9092,ec2-54-236-47-54.compute-
1.amazonaws.com:9092 --topic PURCHASE
```

13. Try to stop one Kafka server from the cluster and recheck the topic configuration with:

```
/usr/local/kafka/bin/kafka-topics.sh --describe --bootstrap-server ec2-54-236-47-54.compute-
1.amazonaws.com:9092 --topic PURCHASE
```

14. Restart the Kafka server and execute the kafka-leader-election to refresh the topic' cluster configuration:

```
/usr/local/kafka/bin/kafka-leader-election.sh --bootstrap-server ec2-54-90-57-82.compute-
1.amazonaws.com:9092,ec2-54-173-171-63.compute-1.amazonaws.com:9092,ec2-54-236-47-54.compute-
1.amazonaws.com:9092 --election-type preferred --all-topic-partitions
```

Hints: if needed the Kafka server logs are located at: /usr/local/kafka/logs/kafkaServer.out where you can see the execution logging.

If you are using already used kafka servers, to make them as a cluster, you may need to remove all old data. Simply remove the /tmp/kafka-logs directory, and recreate it empty.

P. Broker failure test

- Launch the cluster with 2 brokers
- Create a new topic with 4 partitions

```
sudo /usr/local/kafka/bin/kafka-topics.sh --create --bootstrap-server <YourIP_or_DNS>:9092 -replication-
factor 2 --partitions 4 --topic events

Created topic "events".
```

- Check who are the leaders:

```
/usr/local/kafka/bin/kafka-topics.sh --describe --bootstrap-server <YourIP_or_DNS>:9092 --topic events

Topic:events    PartitionCount:4        ReplicationFactor:2     Configs:
        Topic: events   Partition: 0    Leader: 0       Replicas: 0,1   Isr: 0,1
        Topic: events   Partition: 1    Leader: 1       Replicas: 1,0   Isr: 1,0
        Topic: events   Partition: 2    Leader: 0       Replicas: 0,1   Isr: 0,1
        Topic: events   Partition: 3    Leader: 1       Replicas: 1,0   Isr: 1,0
```

- Produce message to the topic

```
/usr/local/kafka/bin/kafka-console-producer.sh --broker-list <YourIP_or_DNS>:9093, <YourIP_or_DNS>:9094 --
topic events
```

- Consume message from the topic <*start two different consumer sessions*>

```
/usr/local/kafka/bin/kafka-console-consumer.sh --bootstrap-server <YourIP_or_DNS>:9093, <YourIP_or_DNS>:9094
--topic events --group group-id-teste-cluster
```

- Check that messages are being balanced between the two consumers

- Stop broker id 0 with the following command:

```
sudo kill `ps aux | grep java | grep server-0.properties | tr -s " " | cut -d " " -f2`
```

- Check the change in the leaders of the cluster

```
/usr/local/kafka/bin/kafka-topics.sh --describe --bootstrap-server <YourIP_or_DNS>:9092 --topic events
```

```
Topic:events    PartitionCount:4        ReplicationFactor:2     Configs:
        Topic: events   Partition: 0    Leader: 1       Replicas: 0,1   Isr: 1
        Topic: events   Partition: 1    Leader: 1       Replicas: 1,0   Isr: 1
        Topic: events   Partition: 2    Leader: 1       Replicas: 0,1   Isr: 1
        Topic: events   Partition: 3    Leader: 1       Replicas: 1,0   Isr: 1
```

- Check that messages are still being balanced between the two consumers

Appendix: URLs with other resources

- https://github.com/confluentinc/confluent-kafka-python/issues/187
- http://cloudurable.com/blog/kafka-tutorial-kafka-failover-kafka-cluster/index.html
- if kafka brokers IDs are corrupted: https://stackoverflow.com/questions/59592518/kafka-broker-doesnt-find-cluster-id-and-creates-new-one-after-docker-restart