Deployment Guide – Kafka Transport for WSO2 ESB

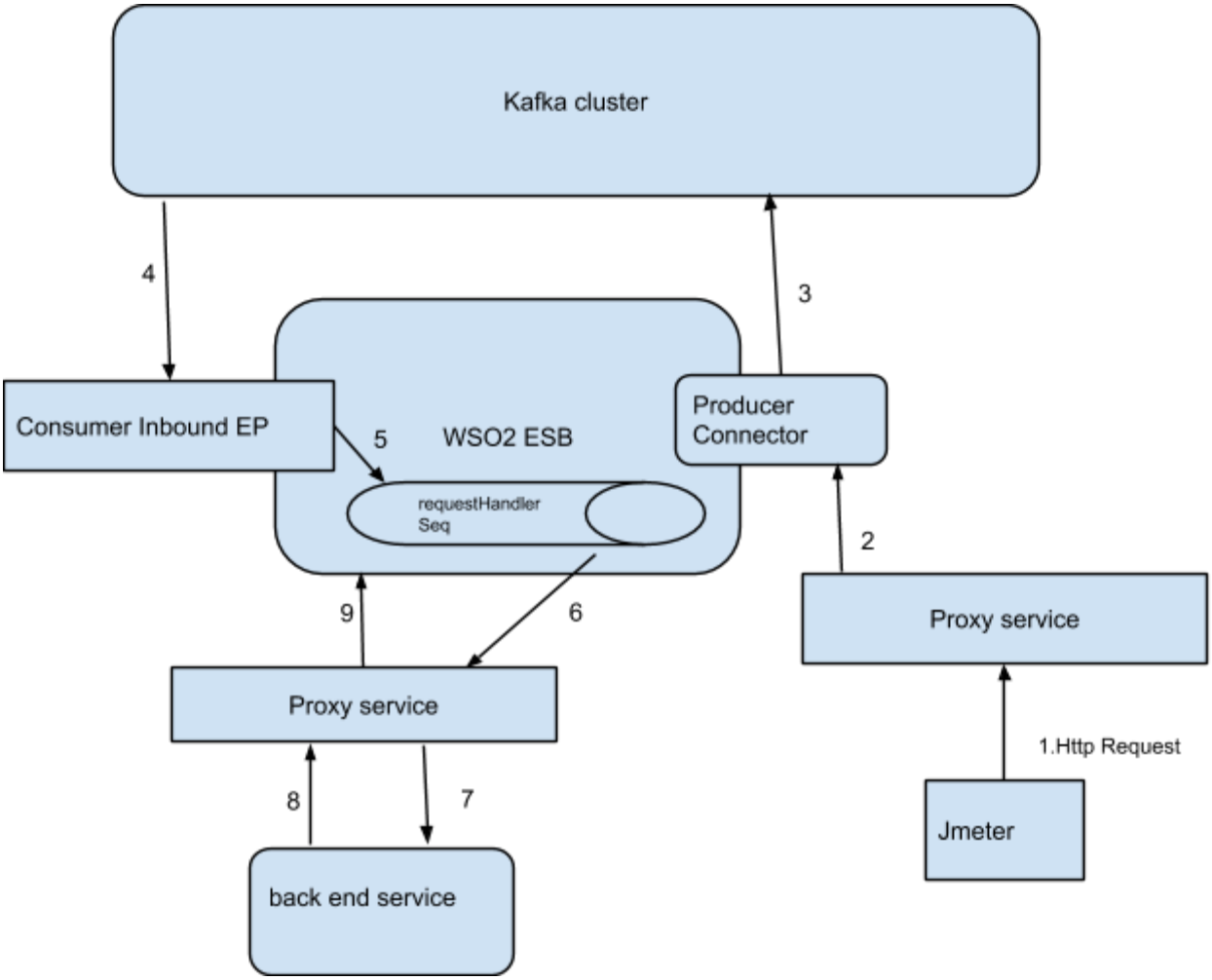

Fig 01: Deployement Diagram

1. Start Kafka Services

- Download apache.kafka distribution form here

  Extract and goto kafka home

  >tar -xzf kafka_2.9.2-0.8.1.1.tgz

>cd kafka_2.9.2-0.8.1.1

- Start zookeeper

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

- Start server

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

Please refer https://kafka.apache.org/documentation.html#introduction

2. **Deploy Kafka Inbound EP into ESB**

- Create following directory structure in the ESB_HOME

>ESB_HOME/repository/deployment/server/synapse-config/default/inbound-endpoints/

- Create MyKAFKAListenerEP.xml inside the inbound-endpoints/ directory

- Copy the following configurations and Save.

```
<inboundEndpoint xmlns="http://ws.apache.org/ns/synapse"
name="MyKAFKAListenerEP"
                    protocol="kafka"
                    interval="1000" suspend="false"
sequence ="requestHandlerSeq" onError="inFaulte">
 <parameters>
<parameter name="zookeeper.connect">localhost:2181</parameter>
<parameter name="group.id">test-group</parameter>
<parameter name="zookeeper.session.timeout.ms">500</parameter>
<parameter name="zookeeper.sync.time.ms">250</parameter>
<parameter name="auto.commit.interval.ms">1000</parameter>
<parameter name="auto.offset.reset">smallest</parameter>
<parameter name="thread.count">1</parameter>
 <parameter name="topics">test2</parameter>
<parameter name="content.type">application/octet-stream</parameter>
<parameter name="topic.filter">test</parameter>
<parameter name="filter.from.whitelist">false</parameter>
```

<parameter name="consumer.type">highlevel</parameter>
<parameter name="simple.max.messages.to.read">5</parameter>
<parameter name="simple.topic">test</parameter>
<parameter name="simple.brokers">localhost</parameter>
<parameter name="simple.port">9092</parameter>
<parameter name="simple.partition">0</parameter>
</parameters>
</inboundEndpoint>

- Copy following jar files to ESB_HOME/repository/components/lib.
  kafka_2.9.2-0.8.1.1.jar
  scala-library-2.9.2.jar
  zkclient-0.3.jar
  zookeeper-3.3.4.jar
  metrics-core-2.2.0.jar
  slf4j-api-1.7.2.jar

- Starts ESB
  ESB_HOME >sh bin/wso2server.sh
  Create sequence named requestHandlerSeq

  Sample for requestHandlerSeq:
  <sequence xmlns="http://ws.apache.org/ns/synapse" name="requestHandlerSeq" onError="inFaulte" trace="enable">
  <log level="full"></log>
  <send>
  <endpoint>
  <address http://ishara-ThinkPad-T540p:8280/services/receiver"></address>
  </endpoint>
  </send>
  <drop></drop>
  </sequence>

- Restart ESB

Inbound Endpoint supports two kafka APIs:

1. Highlevel Consumer API

2. Simple Consumer API

Highlevel Consumer API

To enable Highlevel Consumer API,
<parameter name="consumer.type">highlevel</parameter> should be enabled.

If it needs to be consumed by topic, *<parameter name="topics">test2</parameter>* should be enabled.

If it needs to be consumed by topic filters, *<parameter name="topics">test2</parameter>* should be removed and two properties *<parameter name="topic.filter">test</parameter>* *<parameter name="filter.from.whitelist">false</parameter>*
should have been specified.

If filter.from.whitelist = true, messages will be consumed from whitelist(include) where as if it is false then will be consumed from blacklist(exclude).

Consumer configurations specifies in
https://kafka.apache.org/documentation.html#consumerconfigs can be added to parameters section as a new parameter if required.

For most of the applications highlevel Consumer API is good enough.This can be used to request messages using topic/s or topic filters.

Messages can be filtered either from whitelist or blacklist.It can be configured as
<parameter name="filter.from.whitelist">true</parameter>

or

<parameter name="filter.from.whitelist">false</parameter> respectively.
Simple Consumer API

To enable Simple Consumer API,
<parameter name="consumer.type">simple</parameter> should be enabled.

Using simple consumer API, it is possible to request messages from a specific brocker and from a specific partition.

Sample configuration:

brocker host ip and port
<parameter name="simple.brokers">localhost</parameter>
<parameter name="simple.port">9092</parameter>

partition
<parameter name="simple.partition">0</parameter>

no. of messages to read
<parameter name="simple.max.messages.to.read">5</parameter>

topic
<parameter name="simple.topic">test</parameter>

3. **Kafka Producer Connector into ESB**

- deploy kafka producer connector in ESB

  Download connector zip file from here
  KafKaProducer-Connector.....

- Create a proxy service in ESB

- Add parameters to proxy

add <kafkaTranskaport.init> element in your configuration before you send the messages to kafka brokers.

<kafkaTransport.init>
<brokerlist>localhost:9092</brokerlist>
<serializationclass>kafka.serializer.StringEncoder</serializationclass>

```
        <requiredacks>1</requiredacks>
        <producertyepe>sync</producertype>
    </kafkaTransport.init>
```

additional configuration of kafka Producer

```
    <kafkaTransport.init>
        <brokerlist>localhost:9092</brokerlist>
        <serializationclass>kafka.serializer.StringEncoder</serializationclass>
        <requiredacks>1</requiredacks>
        <producertype>sync</producertype>
        <partitionerclass>kafka.producer.DefaultPartitioner</partitionerclass>
        <keyserializerclass>kafka.serializer.StringEncoder</keyserializerclass>
        <compressioncodec>none</compressioncodec>
        <compressedtopics>null</compressedtopics>
        <messagesendmaxretries>3</messagesendmaxretries>
        <retrybackoff>100</retrybackoff>
        <refreshinterval>600000</refreshinterval>
        <bufferingmaxmessages>10000</bufferingmaxmessages>
        <batchnomessages>200</batchnomessages>
        <sendbuffersize>102400</sendbuffersize>
    </kafkaTransport.init>
```

Send the message using topic

```
    <kafkaTransport.kafkaproduce-operation>
        <topic>mytopic</topic>
    </kafkaTransport.kafkaproduce-operation>
```

send the message using topic and key

```
    <kafkaTransport.kafkaproduce-operation>
        <topic>mytopic</topic>
        <key>key1</key>
    </kafkaTransport.kafkaproduce-operation>
```