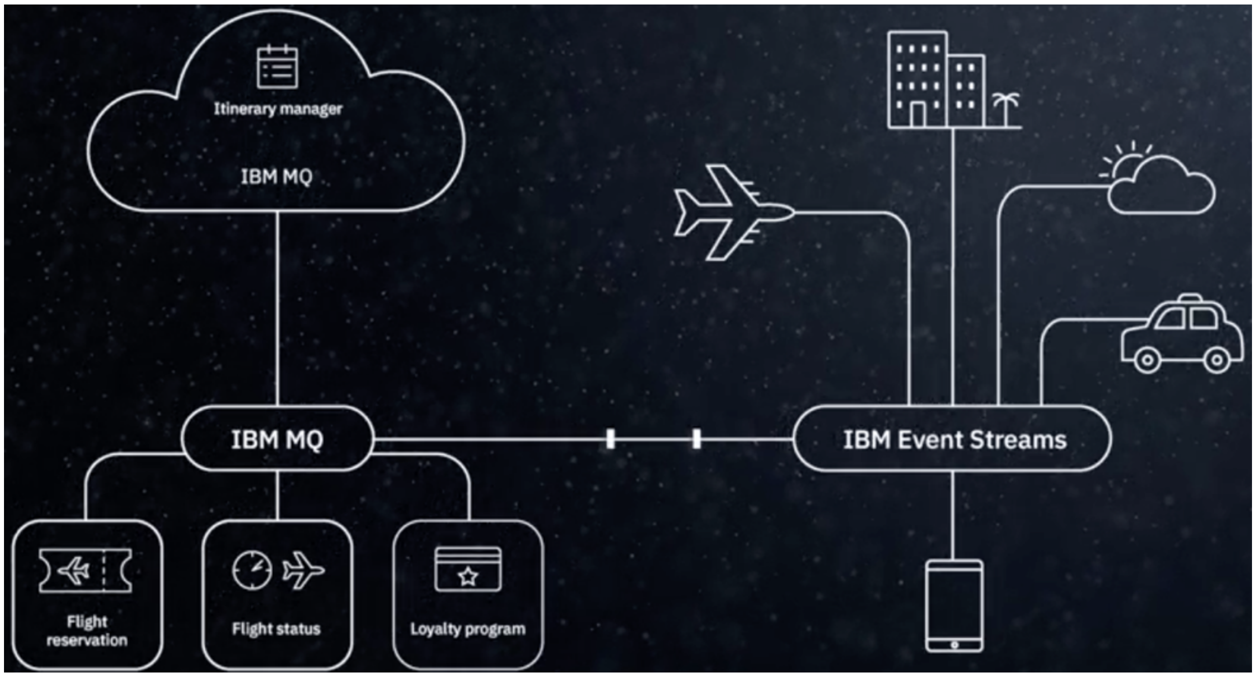
Improve your customer experience by reacting to situations in real time

The most interesting and impactful new applications in an enterprise are the applications that provide new ways of interacting with existing systems by reacting in real time to mission-critical data. Leverage your existing investments, skills and even existing data, and use event-driven techniques to offer more-responsive and more-personalized experiences. IBM Event Streams has supported connectivity to the systems you're already using. By combining the capabilities of IBM Event Streams event streams and message queues, you can combine your transaction data with real time events to create applications and processes. These applications and processes will allow you to react to situations quickly and provide a greater personalized experience.



In this tutorial, you create a bidirectional connection between MQ and Event Streams by creating two message queues and two event stream topics. One is for sending and one for receiving. You then configure the message queue source and sync connectors to connect between the two instances.

In this tutorial, you will explore the following key capabilities:

- Configure MQ to send and receive messages and events
- Configure Event Streams topics
- Configure MQ Source and Sync Connectors.
- Test connectors MQ source and sink (send a message and receive events).

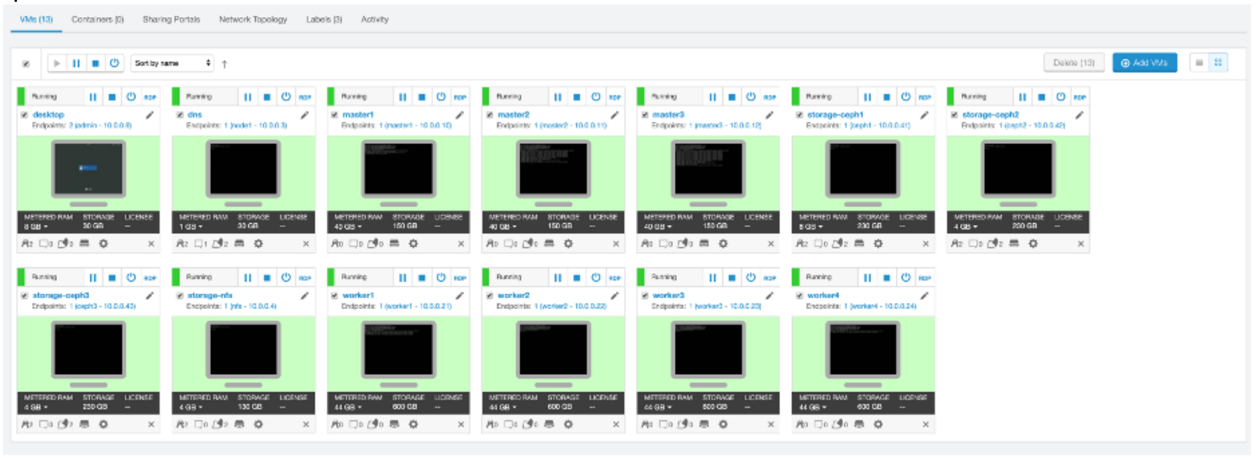
Task 1- Start the environment

Because this is a new deployment of the Cloud Pak for Integration that uses Red Hat OpenShift, you need to execute some steps to prepare the environment. Initial setup steps are only needed for a fresh installation of the Cloud Pak. They do not need to be repeated.

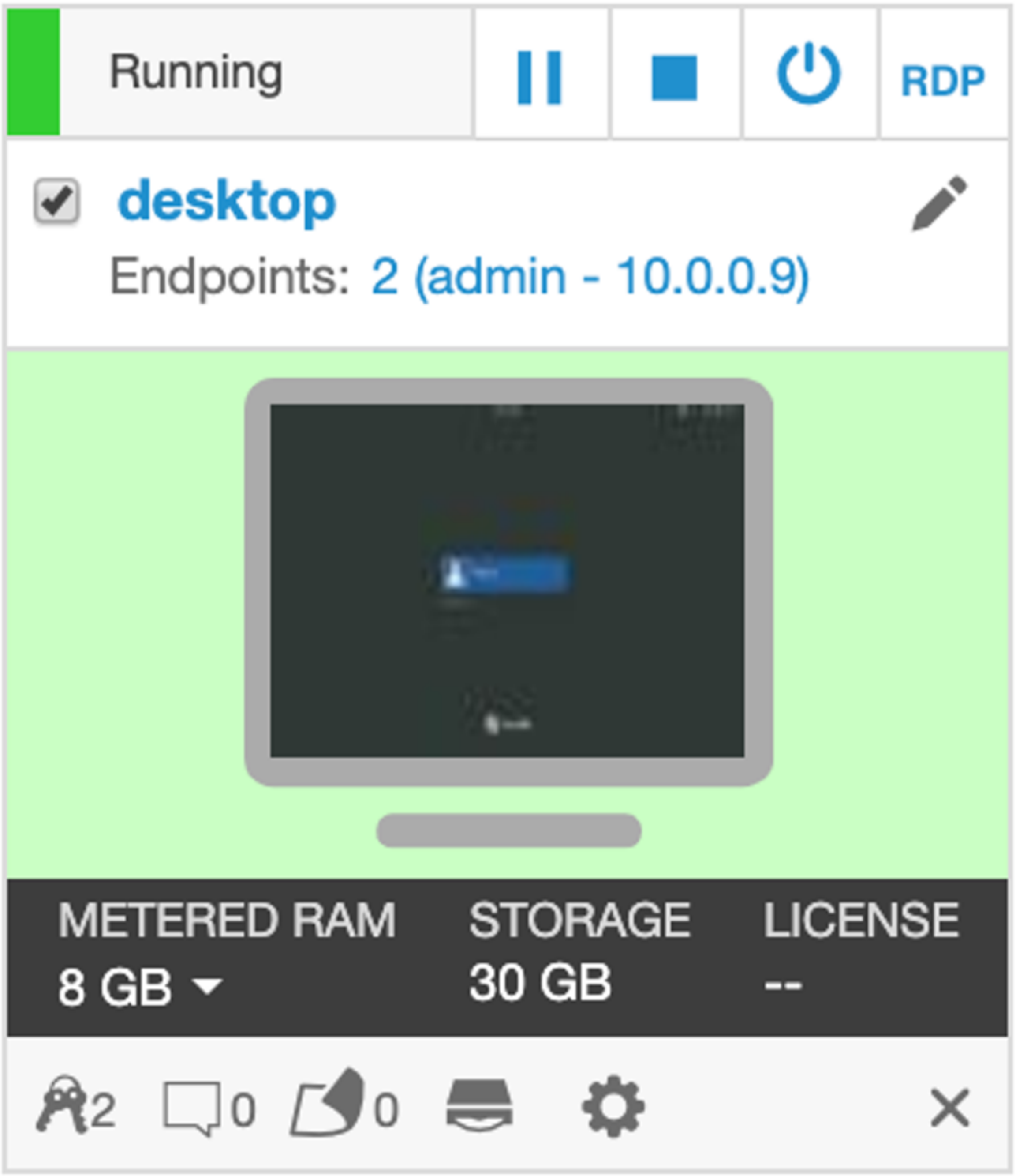
All work for this lab is done on the desktop Machine. Open the **desktop machine** virtual machine by clicking the tile.

Start the Environment

1. When you open your reservation link, the environment should be up and running already and you can skip to step 3. If it is not running, proceed to Step 2.



2. Click the run button as shown below to start the virtual machine environment that is used for this lab.
3. When the virtual machine starts, click the **Desktop Machine screen image** to start your lab exercise.

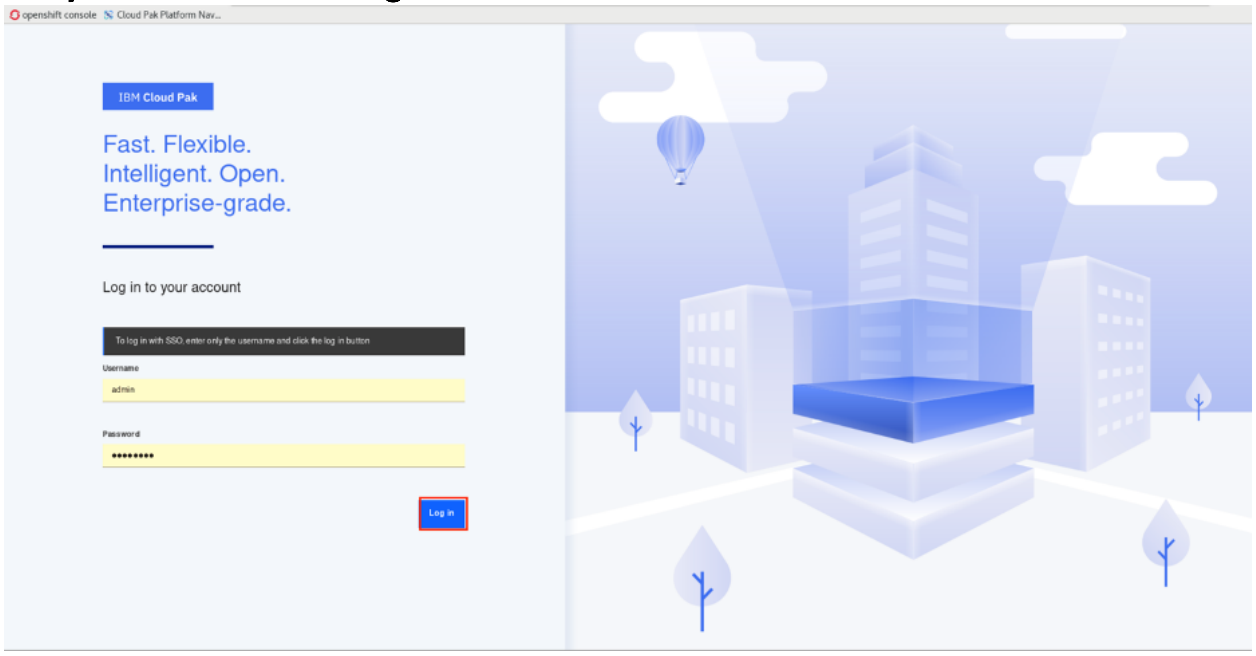


4. When the virtual machine starts, click the **desktop Machine screen image** to start your lab exercise.
5. Log in to the Linux desktop with userid: **ibmuser** and password: "**engageibm".**

Task 2 – Configuring MQ

In this task, you work with MQ Console, create two queues (MQTOEVENT and FROMEVENT),

1. In your browser, click the **IBM Cloud Platform Navigator** bookmark.You might need to log in to IBM Cloud Pak. The username and Password are already cached. Click **Log in.**



Fast. Flexible.
Intelligent. Open.
Enterprise-grade.

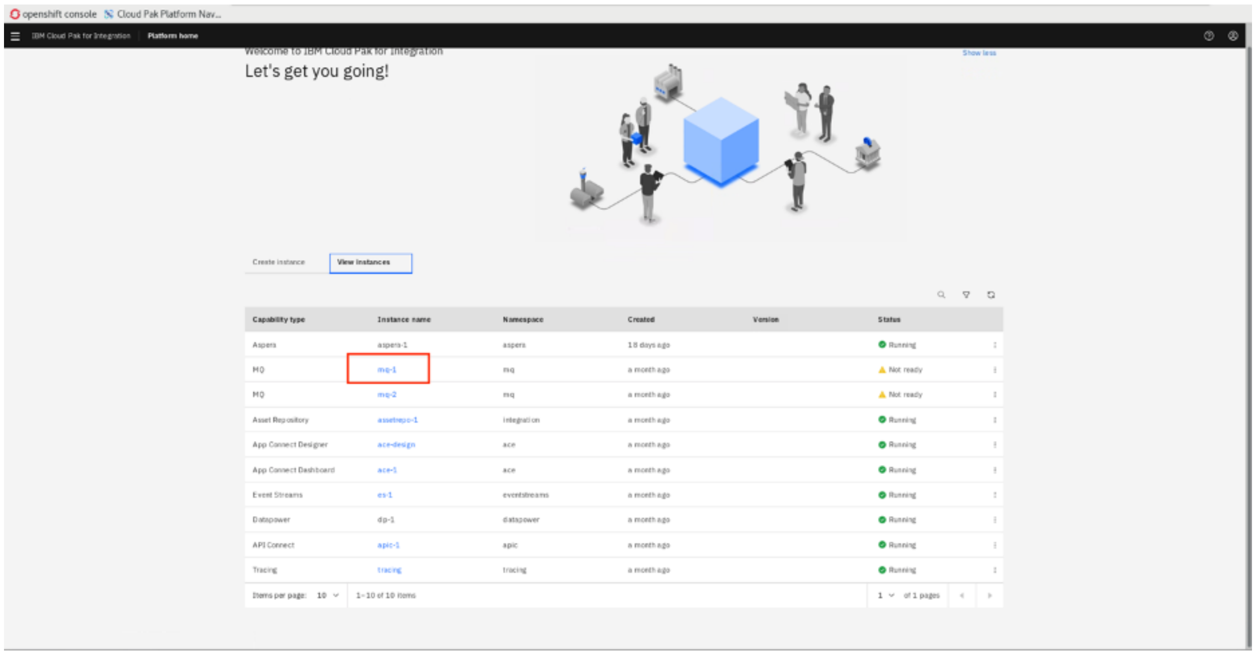
Log in to your account

To log in with SSO, enter only the username and click the log in button

Username: admin

Password: •••••••••

2. In the welcome to IBM Cloud Pak for Integration page, click **View Instances** and on the Cloud Pak for Integration, locate the **MQ** service and click **mq-1.**



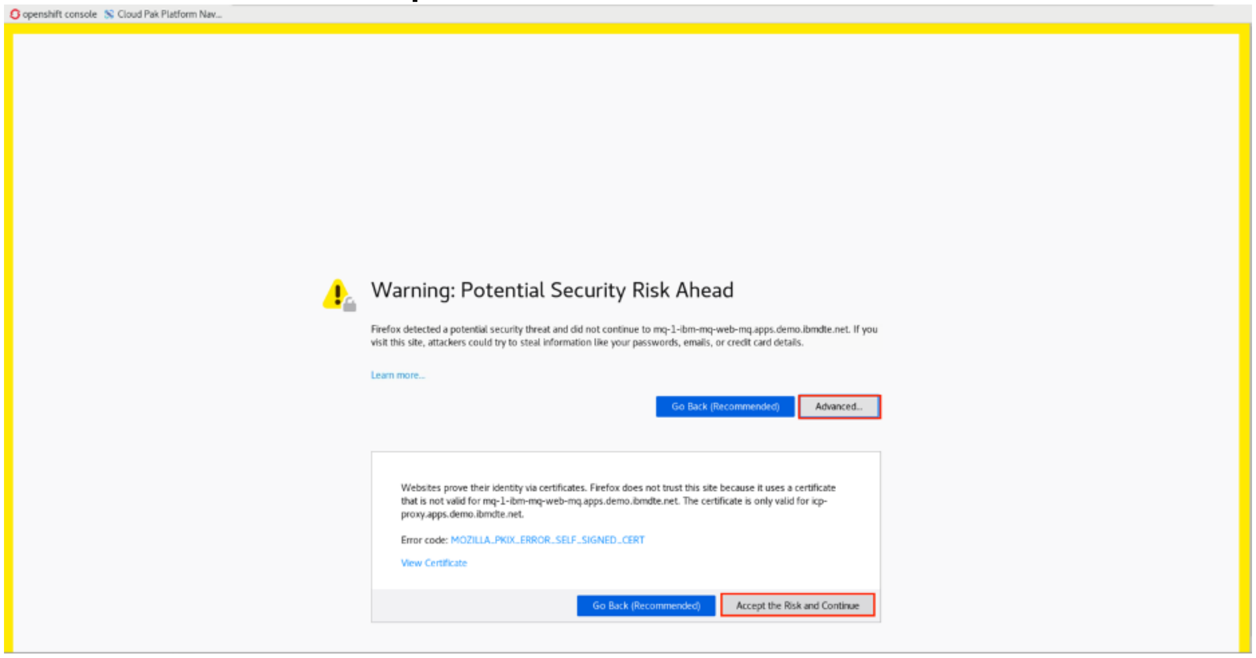
Welcome to IBM Cloud Pak for Integration

Let's get you going!

Capability type	Instance name	Namespace	Created	Version	Status
Aspera	aspera-1	aspera	18 days ago		Running
MQ	mq-1	mq	a month ago		Not ready
MQ	mq-2	mq	a month ago		Not ready
Asset Repository	assetrepo-1	integration	a month ago		Running
App Connect Designer	ace-design	ace	a month ago		Running
App Connect Dashboard	ace-1	ace	a month ago		Running
Event Streams	es-1	eventstreams	a month ago		Running
Datapower	dp-1	datapower	a month ago		Running
API Connect	apic-1	apic	a month ago		Running
Tracing	tracing	tracing	a month ago		Running

Items per page: 10   1–10 of 10 items   1   of 1 pages

3. Firefox warns you about a potential security risk.
   Click **Advanced** then **accept the risk and Continue**.



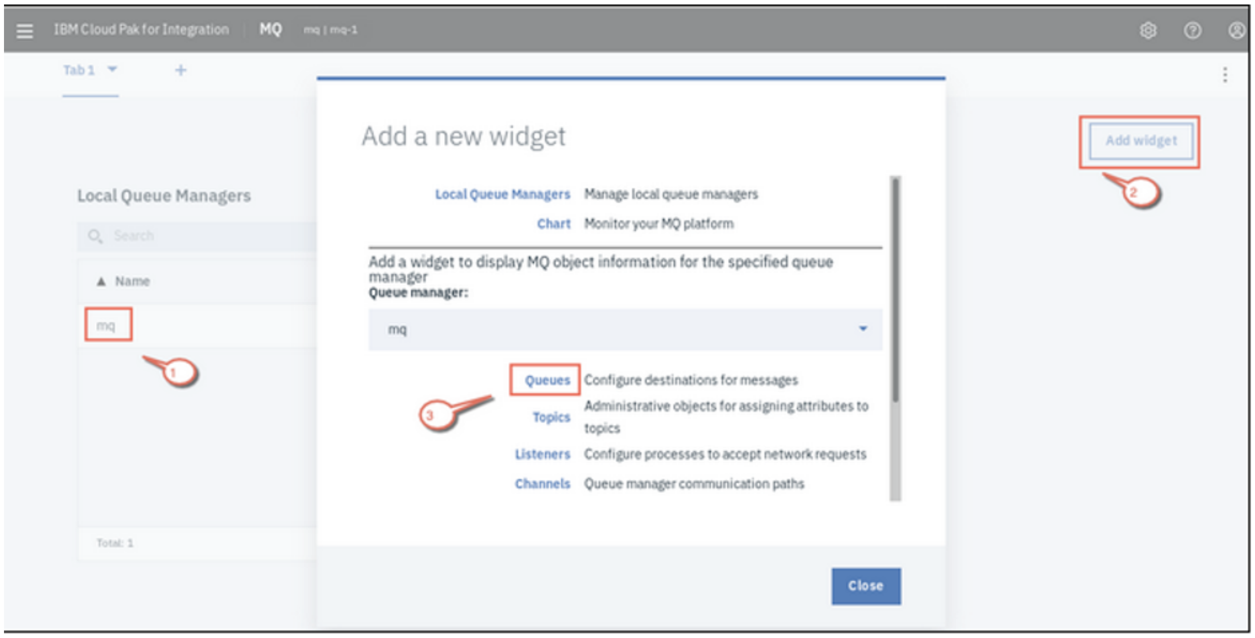
Warning: Potential Security Risk Ahead

Firefox detected a potential security threat and did not continue to mq-1-ibm-mq-web-mq.apps.demo.ibmdte.net. If you visit this site, attackers could try to steal information like your passwords, emails, or credit card details.

Websites prove their identity via certificates. Firefox does not trust this site because it uses a certificate that is not valid for mq-1-ibm-mq-web-mq.apps.demo.ibmdte.net. The certificate is only valid for icp-proxy.apps.demo.ibmdte.net.

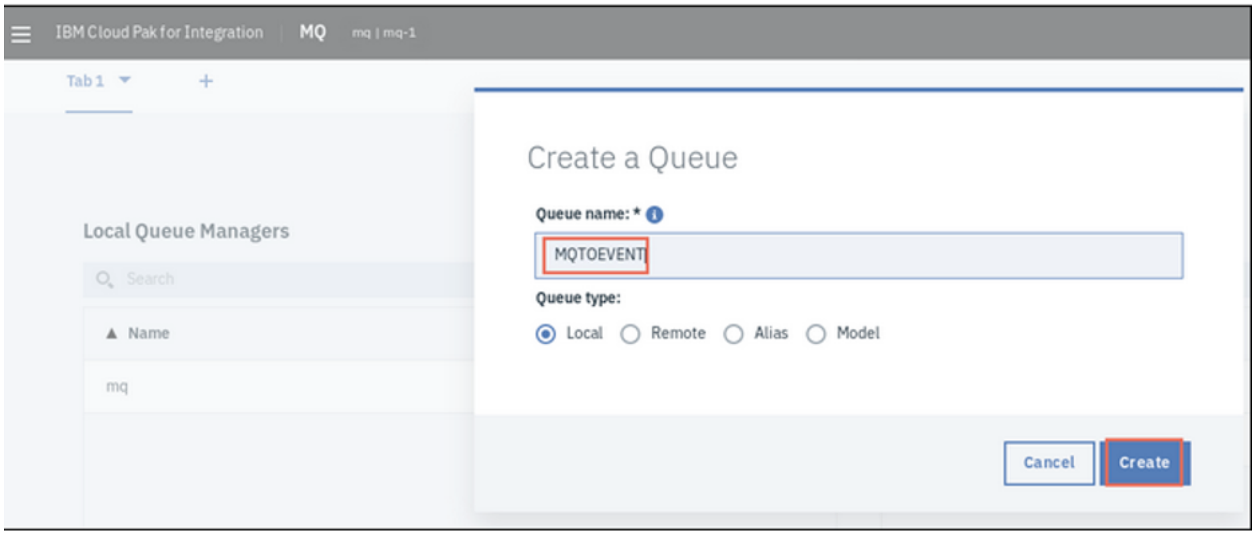
Error code: MOZILLA_PKIX_ERROR_SELF_SIGNED_CERT

4. Since this is the first time to open the mq-1 instance you might need to log in. Click **Log in.**
5. The IBM MQ console opens and you see the queue manager mq running. Click **Add Widget** then click **Queue** to display the queues.


IBM Cloud Pak for Integration
MQ
mq | mq-1
Tab 1
Add a new widget
Local Queue Managers
Manage local queue managers
Chart
Monitor your MQ platform
Add a widget to display MQ object information for the specified queue manager
Queue manager:
mq
Queues
Configure destinations for messages
Topics
Administrative objects for assigning attributes to topics
Listeners
Configure processes to accept network requests
Channels
Queue manager communication paths
Close
Add widget
Local Queue Managers
Search
Name
mq
Total: 1
1
2
3

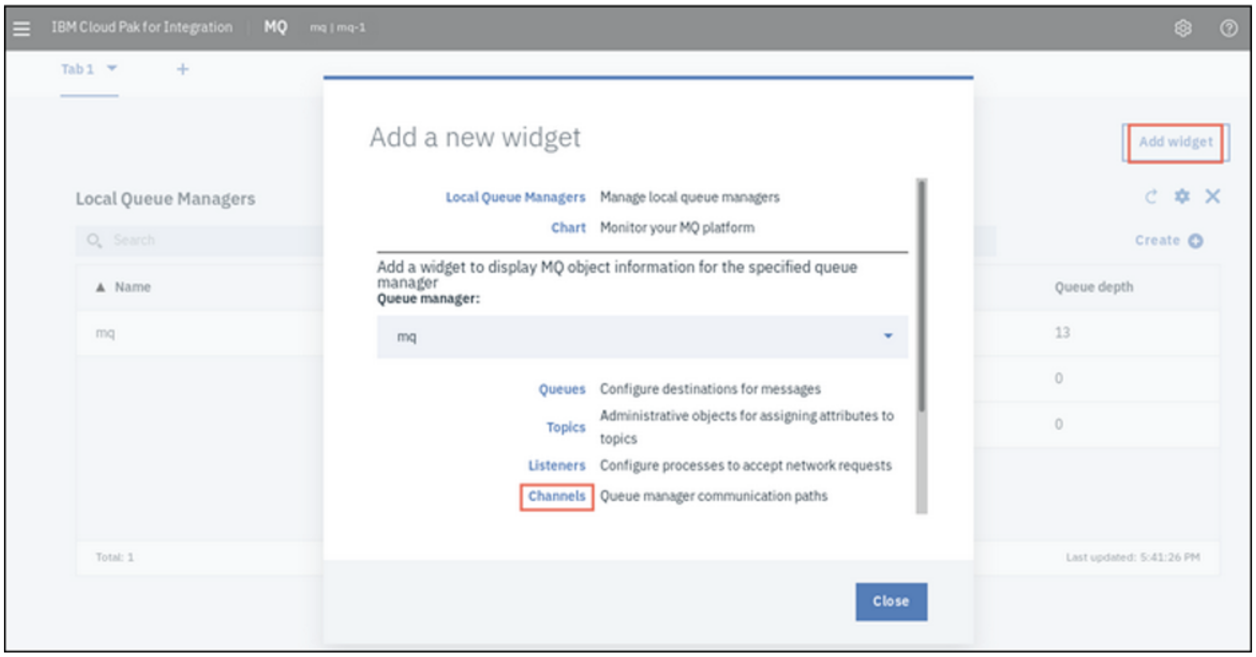
6. Click **Create** (**+**) and enter the name MQTOEVENT. Accept the default queue type as **Local** and click **Create.**

Note: MQTOEVENT – This queue is the source for the IBM MQ source connector.



7. Repeat the process to add a queue EVENTTOMQ

Note: EVENTTOMQ – This queue is the source for the IBM MQ sink connector.



Add a new widget

Local Queue Managers	Manage local queue managers
Chart	Monitor your MQ platform

Add a widget to display MQ object information for the specified queue manager
Queue manager:
mq

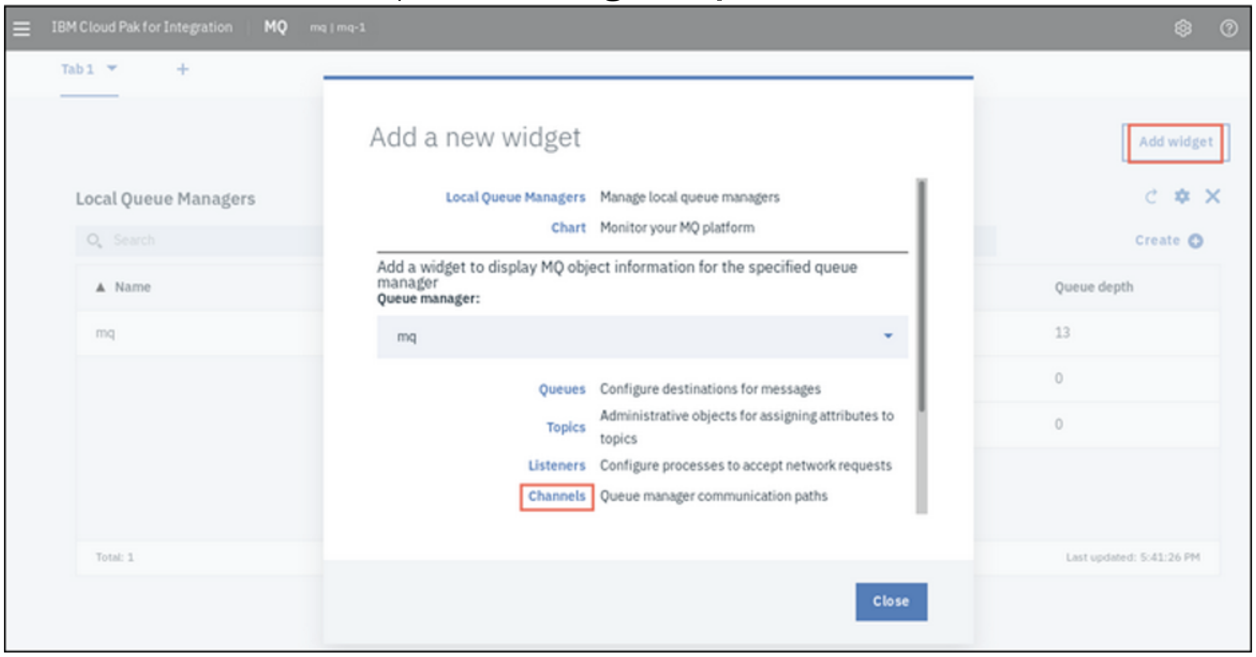
Queues	Configure destinations for messages
Topics	Administrative objects for assigning attributes to topics
Listeners	Configure processes to accept network requests
Channels	Queue manager communication paths

Local Queue Managers
Search

Name	Queue depth
mq	13
	0
	0

Total: 1
Last updated: 5:41:26 PM

8. Click Add widget again and select **Channels.**
   Note: Notice that is already a predefined channel DEF.SVRCONN of type server-connection that has been preconfigured for connecting MQ clients to the MQ queue manager **mq.**



Add a new widget

Local Queue Managers	Manage local queue managers
Chart	Monitor your MQ platform

Add a widget to display MQ object information for the specified queue manager
Queue manager:
mq

Queues	Configure destinations for messages
Topics	Administrative objects for assigning attributes to topics
Listeners	Configure processes to accept network requests
Channels	Queue manager communication paths

Local Queue Managers
Search

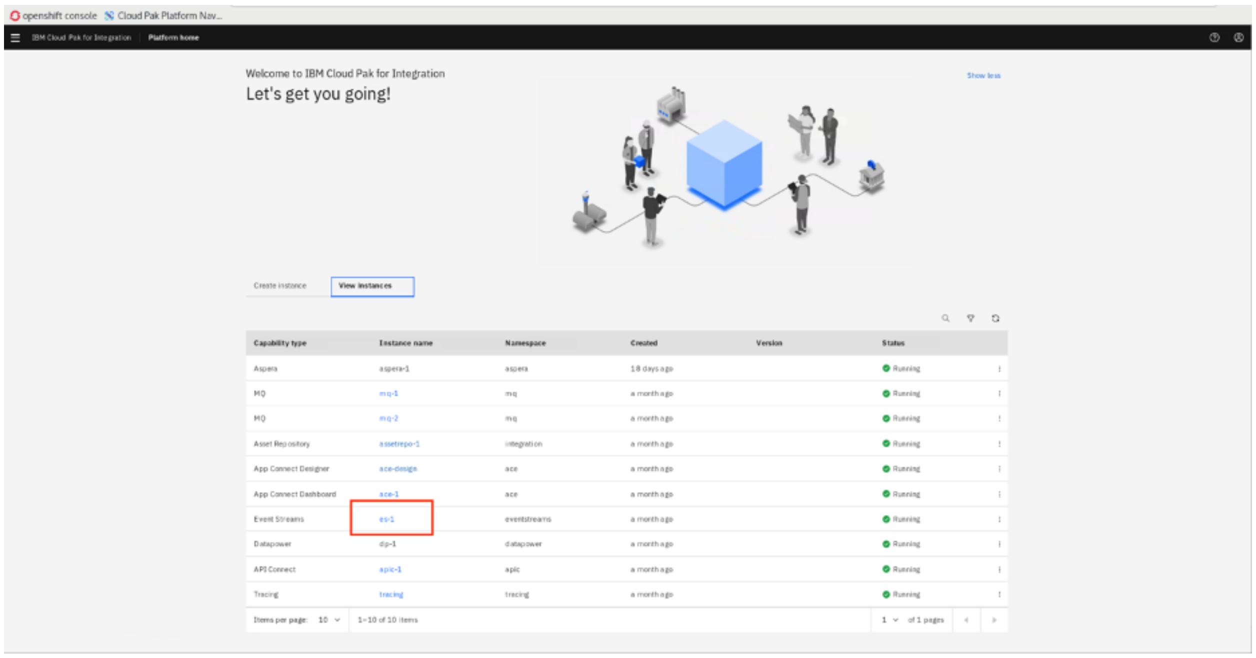
Name	Queue depth
mq	13
	0
	0

Total: 1
Last updated: 5:41:26 PM

Task 3 – Configuring Event Streams
For this lab you will be running Kafka Connect in stand-alone mode. When running in stand-alone mode, Kafka Connect uses a local file

for store configuration, current offsets, and status. Now that you are familiar with topics and creating them from the previous labs, you need to create two new topics for running the MQ source and sink connector.

1. Open a browser and click **IBM Cloud Pak Platform Navigator**. Select **View instances** and click **es-1.**



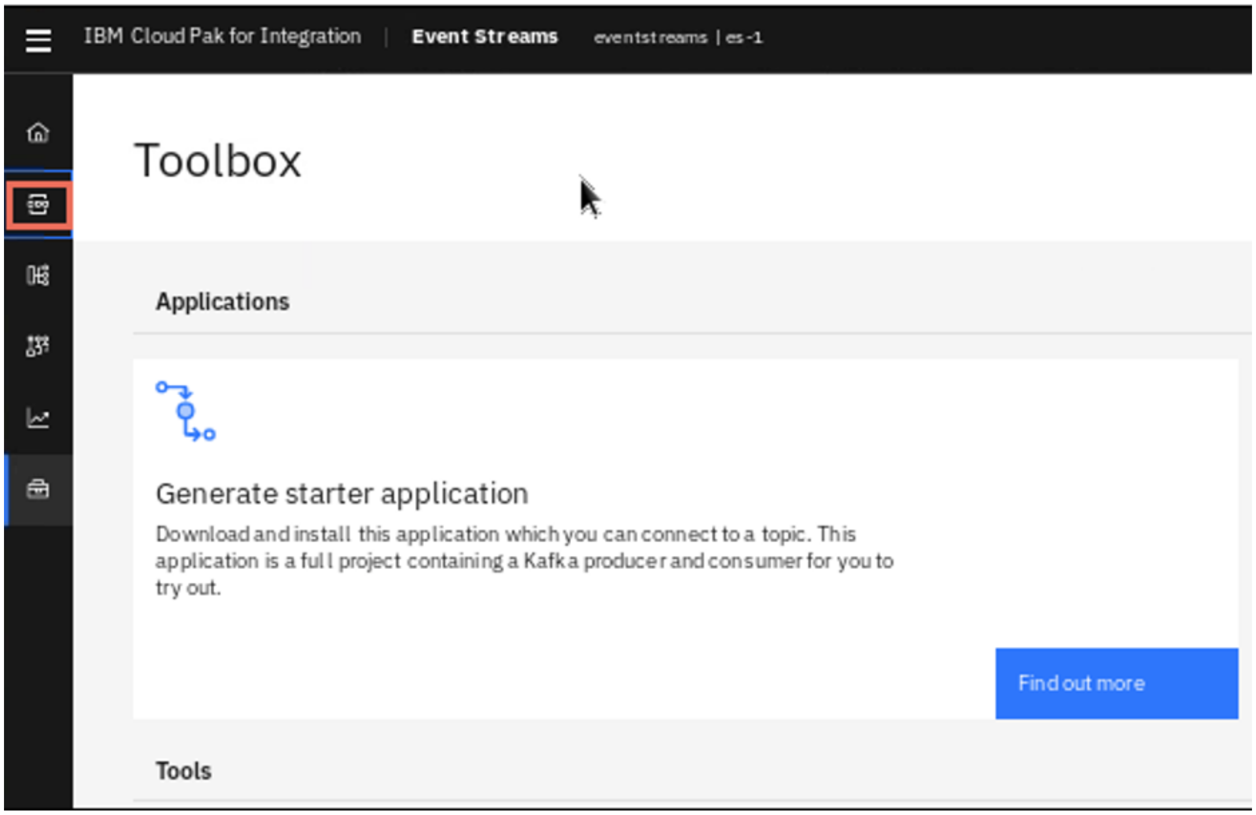
Welcome to IBM Cloud Pak for Integration

Let's get you going!

Capability type	Instance name	Namespace	Created	Version	Status
Aspera	aspera-1	aspera	18 days ago		● Running
MQ	mq-1	mq	a month ago		● Running
MQ	mq-2	mq	a month ago		● Running
Asset Repository	assetrepo-1	integration	a month ago		● Running
App Connect Designer	ace-design	ace	a month ago		● Running
App Connect Dashboard	ace-1	ace	a month ago		● Running
Event Streams	es-1	eventstreams	a month ago		● Running
Datapower	dp-1	datapower	a month ago		● Running
API Connect	apic-1	apic	a month ago		● Running
Tracing	tracing	tracing	a month ago		● Running

Items per page: 10 1–10 of 10 items 1 of 1 pages

2. In the Event Streams UI, click the **Topics** icon on the menu bar.



Toolbox

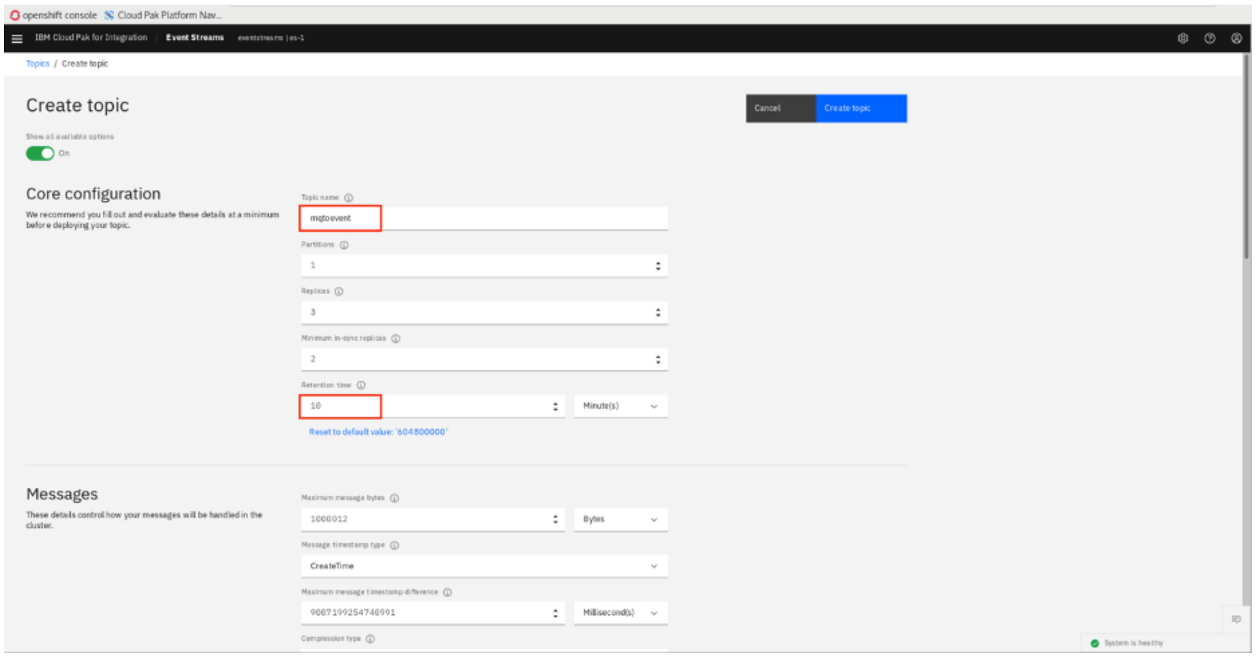
Applications

Generate starter application

Download and install this application which you can connect to a topic. This application is a full project containing a Kafka producer and consumer for you to try out.

3. Click **Create topic**. Turn Show all available options to **On**.
   Enter **mqtoevent** for the topic name.
   Use the following values when creating the topics. Create the following topics: Use the following values when creating the topics**.**

1. **Topic Name** field**,** type: **mqtoevent**
2. **Partitions**: **1** (A partition is an ordered list of messages)
3. **Replicas: 3** (To improve availability, each topic can be replicated onto multiple brokers)
4. **Minimum in-sync replicas: 2** - (To improve availability, each topic can be replicated onto multiple brokers)
5. **Retention time: 10 minutes** - (It Is time messages are retained before being deleted.)
6. Click **Create topic .**



Create topic

Show all available options: On

Core configuration

We recommend you fill out and evaluate these details at a minimum before deploying your topic.

Topic name: mqtoevent

Partitions: 1

Replicas: 3

Minimum in-sync replicas: 2

Retention time: 10 Minute(s)

Reset to default value: '604800000'

Messages

These details control how your messages will be handled in the cluster.

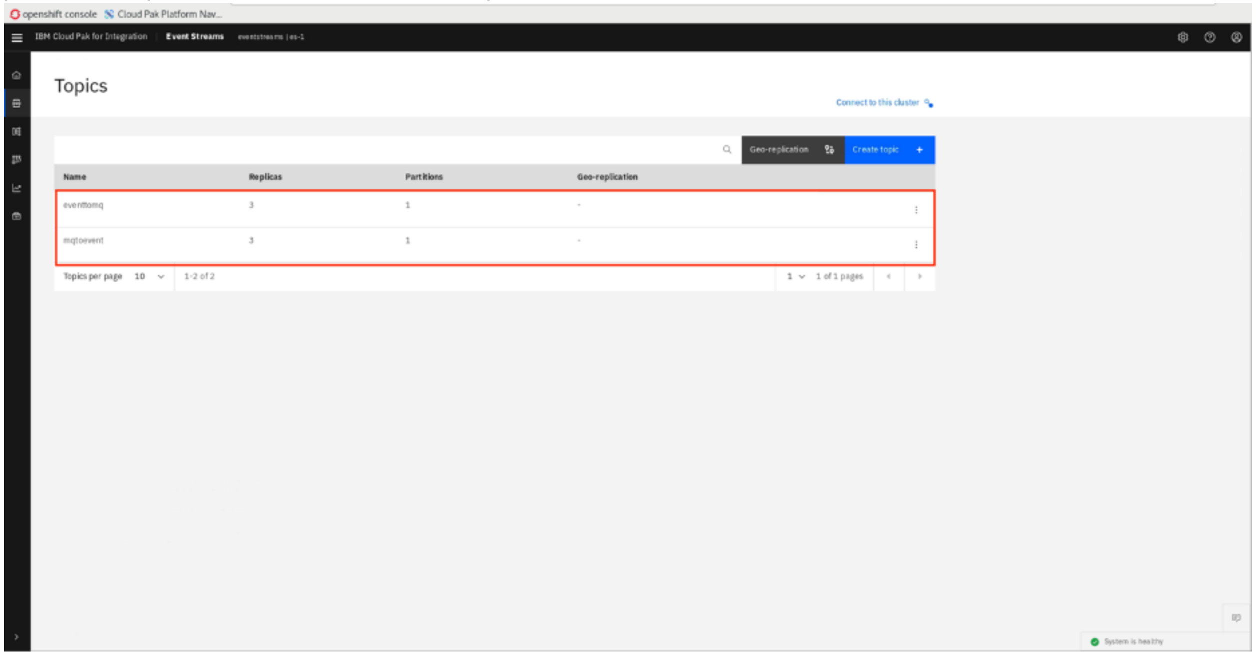
Maximum message bytes: 1000012 Bytes

Message timestamp type: CreateTime

Maximum message timestamp difference: 9007199254740991 Millisecond(s)

Compression type

4. Repeat step 3, and create a topic **eventtomq**.



Topics

Name	Replicas	Partitions	Geo-replication
eventtomq	3	1	-
mqtoevent	3	1	-

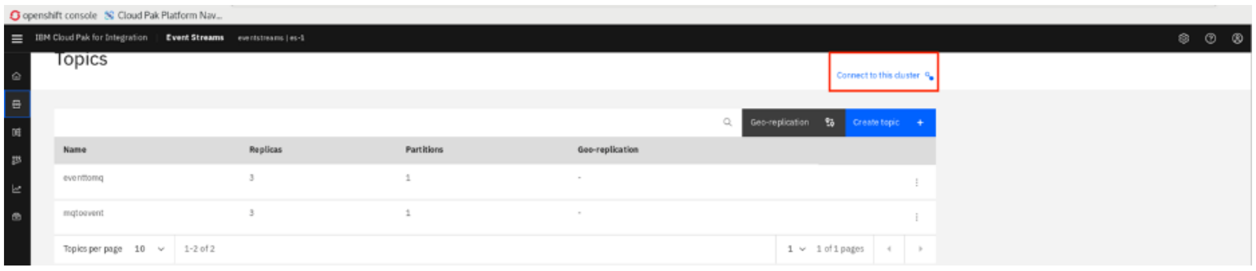
Topics per page 10 1-2 of 2 1 1 of 1 pages

Note: When all available options are **On** you see a great amount of detail for each topic. You only need to edit the Core configuration section. If you turn the switch to Off you need to click **Next** three times before the final pane where you click **Create topic .**

Task 4 - Create API Key and Event Streams certificate (Event Streams Operations)

You need some security parameters to execute Kafka Connectors.

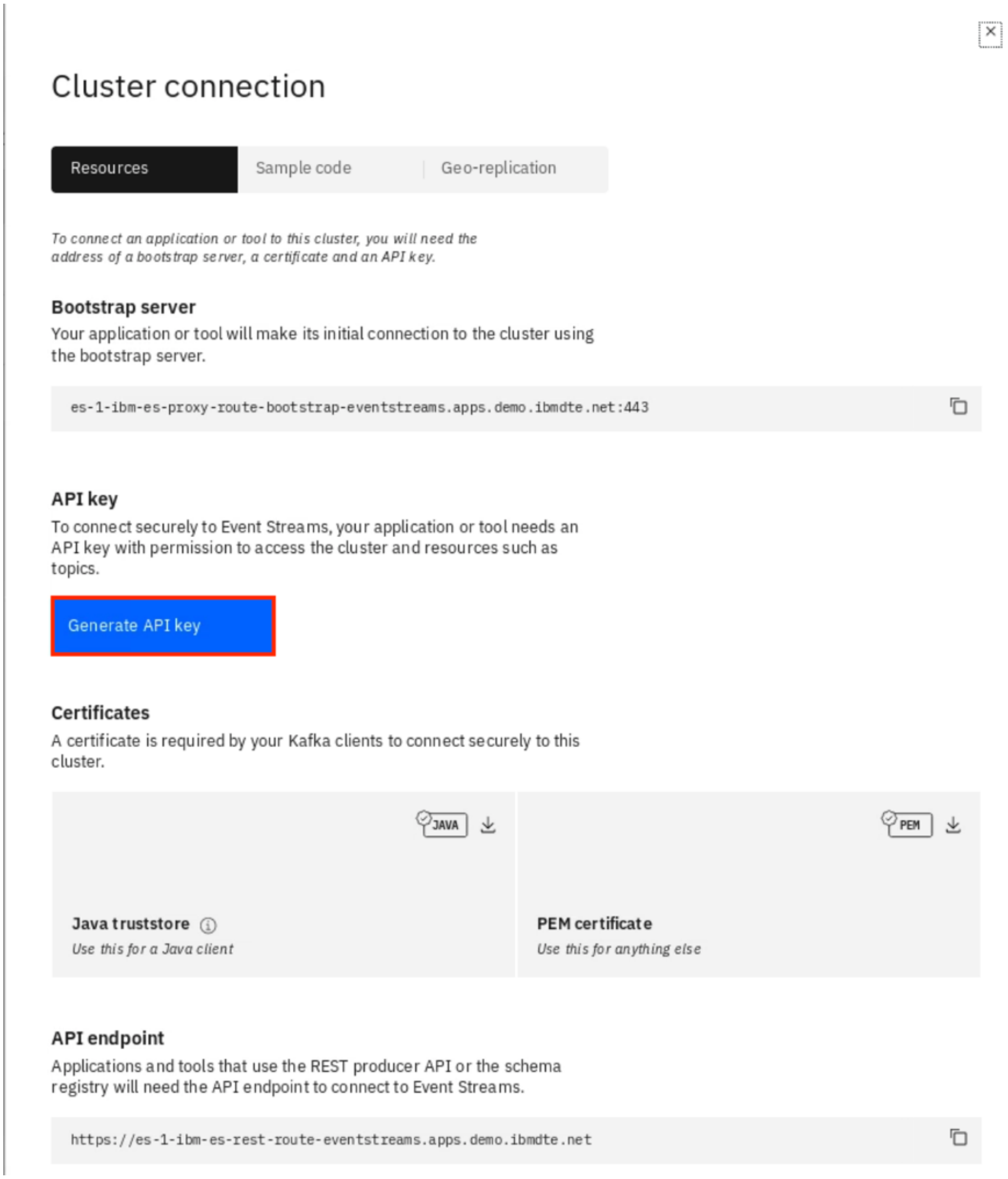
1. In your browser, click the I**BM Cloud Pak for Integration** Bookmark and open **es-1** in **Event Streams** window. Click the **Topics** link and click the **Connect to this cluster** link .



Name	Replicas	Partitions	Geo-replication
eventtomq	3	1	-
mqtoevent	3	1	-

Topics per page 10 1-2 of 2 1 1 of 1 pages

2. In the Cluster connection page, you see Event Streams connection information.
1. Bootstrap server is used to connect an application or tool to this cluster.
2. API Key is to connect with permission to access the cluster and resources such as topics.
3. Certificates are required by the Kafka clients to connect securely to this cluster.
3. Save the Bootstrap server in a file.
1. Right-click the desktop workspace and open a terminal window
2. Create a file using any editor, for example gedit **creds.txt** and copy **es-1-ibm-es-proxy-route-bootstrap-eventstreams.apps.demo.ibmdte.net:443** the bootstrap server.
3. Click **Generate API Key** to create a permission to access the topic.


Cluster connection

To connect an application or tool to this cluster, you will need the address of a bootstrap server, a certificate and an API key.

Bootstrap server

Your application or tool will make its initial connection to the cluster using the bootstrap server.

```
es-1-ibm-es-proxy-route-bootstrap-eventstreams.apps.demo.ibmdte.net:443
```

API key

To connect securely to Event Streams, your application or tool needs an API key with permission to access the cluster and resources such as topics.

Certificates

A certificate is required by your Kafka clients to connect securely to this cluster.

Java truststore
Use this for a Java client

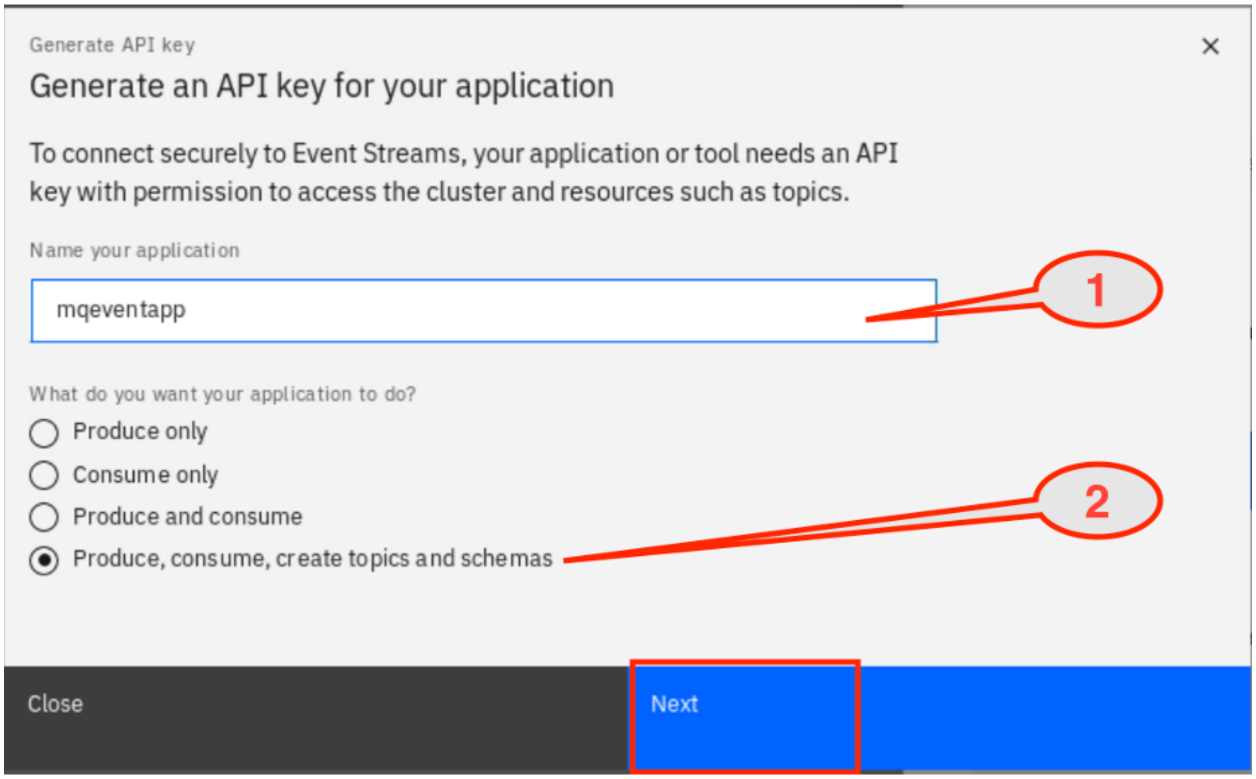
PEM certificate
Use this for anything else

API endpoint

Applications and tools that use the REST producer API or the schema registry will need the API endpoint to connect to Event Streams.

```
https://es-1-ibm-es-rest-route-eventstreams.apps.demo.ibmdte.net
```

4. In the **Generate an API** key for your application, Enter mqeventapp as **Name your application.**
1. Check **Produce, consume, create topics and schemas** box and then click **Next.**


Generate API key

Generate an API key for your application

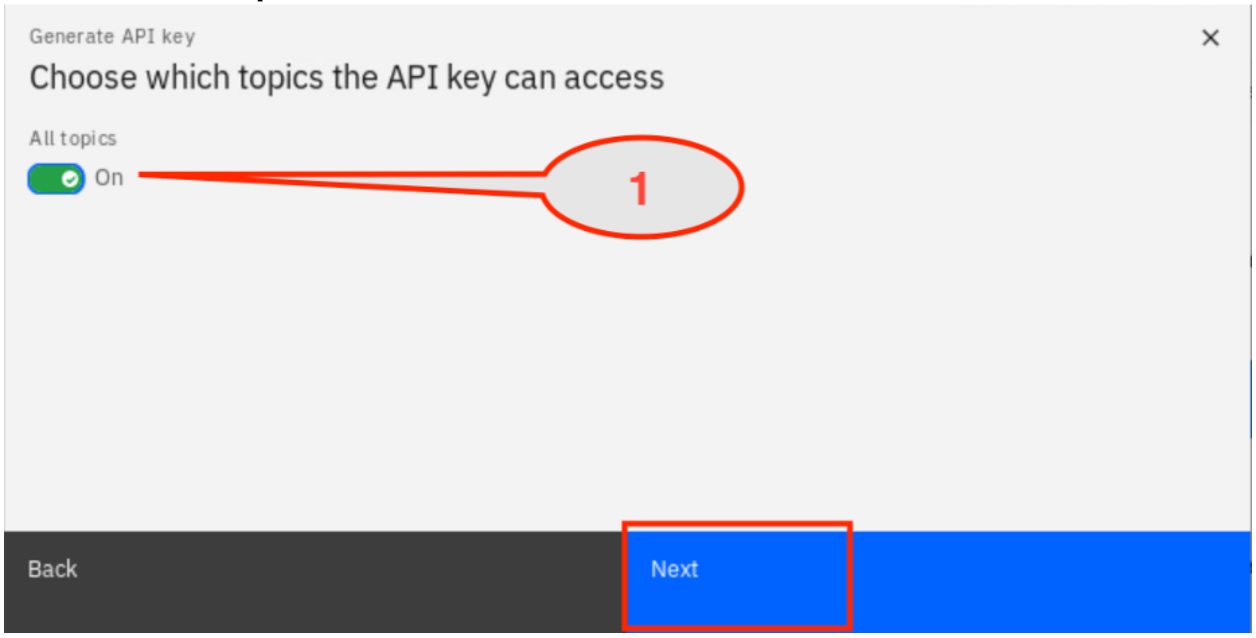
To connect securely to Event Streams, your application or tool needs an API key with permission to access the cluster and resources such as topics.

Name your application: mqeventapp

What do you want your application to do?

- ○ Produce only
- ○ Consume only
- ○ Produce and consume
- ● Produce, consume, create topics and schemas

2. Turn on the **All topics** button and click **Next.**

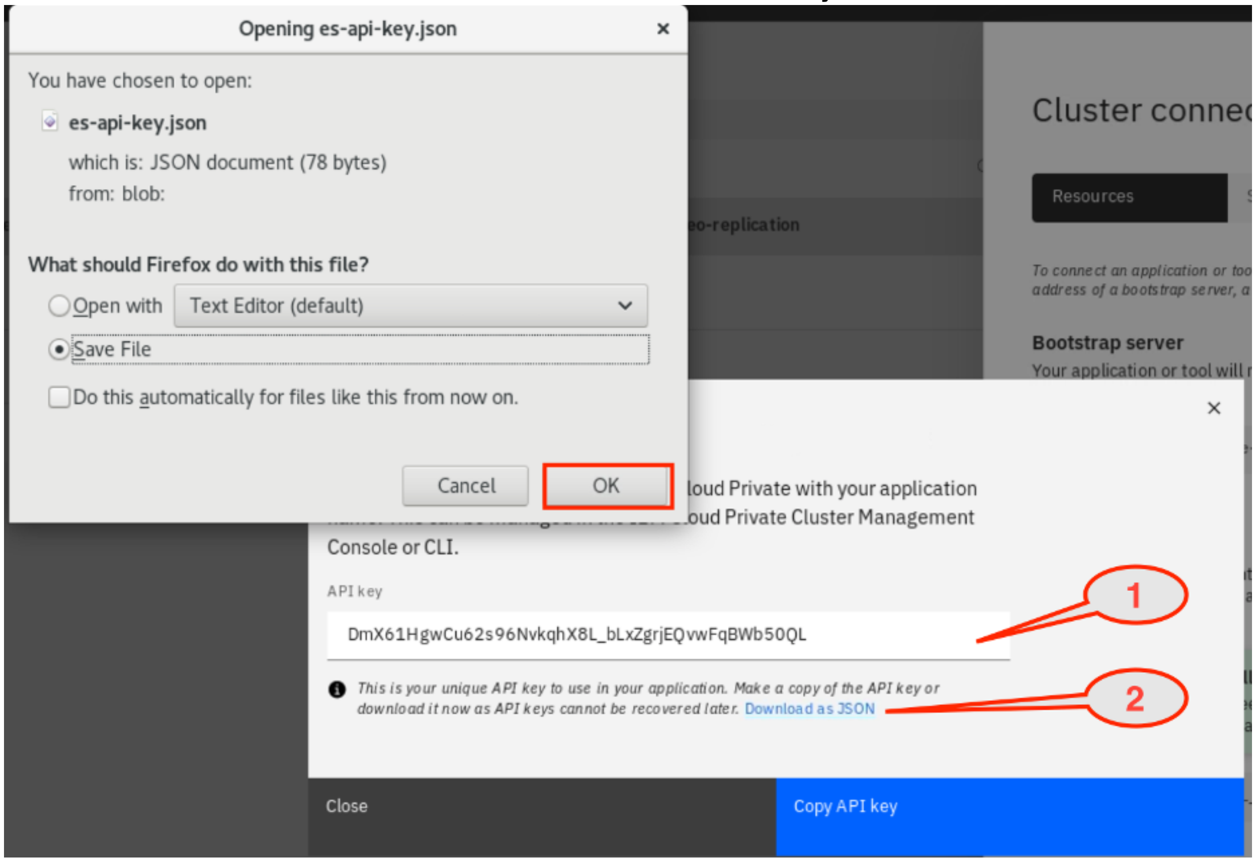

Generate API key

Choose which topics the API key can access

All topics: On

3. Click **Generate API Key .**

5. Event Streams generates an API Key. You can save or download as JSON. Click **Download as a JSON** link. Close the API Key window.


Opening es-api-key.json

You have chosen to open:

es-api-key.json

which is: JSON document (78 bytes)

from: blob:

What should Firefox do with this file?

- [ ] Open with Text Editor (default)
- [x] Save File
- [ ] Do this automatically for files like this from now on.

Cancel | OK

Cluster connec

Resources

To connect an application or too address of a bootstrap server, a

Bootstrap server

Your application or tool will r

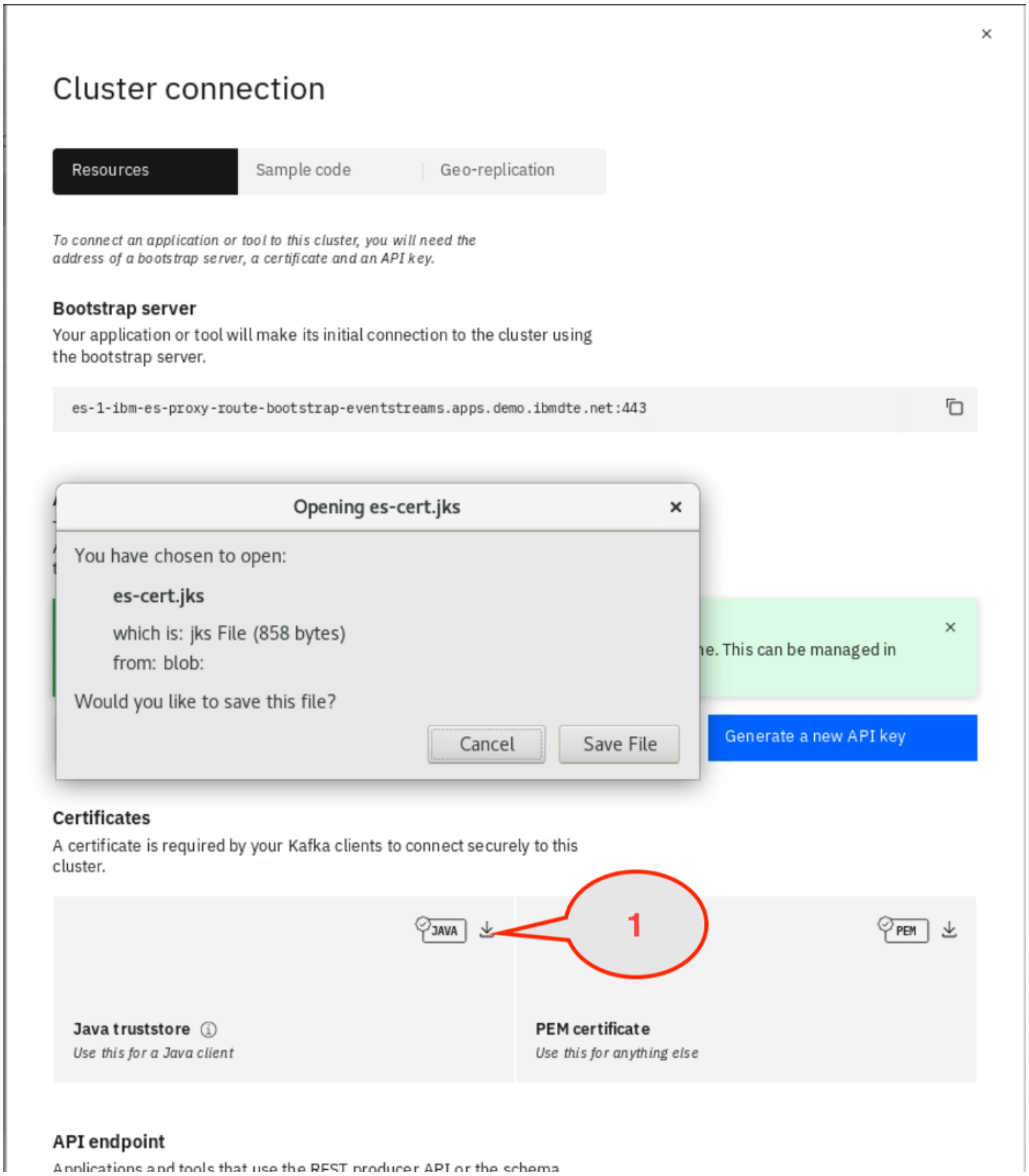
loud Private with your application ... oud Private Cluster Management Console or CLI.

API key

DmX61HgwCu62s96NvkqhX8L_bLxZgrjEQvwFqBWb50QL

This is your unique API key to use in your application. Make a copy of the API key or download it now as API keys cannot be recovered later. Download as JSON

6. In the Certificates, download the truststore certificate. Locate Java truststore, click the download icon and save.


Cluster connection

Resources | Sample code | Geo-replication

To connect an application or tool to this cluster, you will need the address of a bootstrap server, a certificate and an API key.

Bootstrap server

Your application or tool will make its initial connection to the cluster using the bootstrap server.

```
es-1-ibm-es-proxy-route-bootstrap-eventstreams.apps.demo.ibmdte.net:443
```

Opening es-cert.jks

You have chosen to open:

es-cert.jks

which is: jks File (858 bytes)

from: blob:

Would you like to save this file?

Cancel | Save File

e. This can be managed in

Generate a new API key

Certificates

A certificate is required by your Kafka clients to connect securely to this cluster.

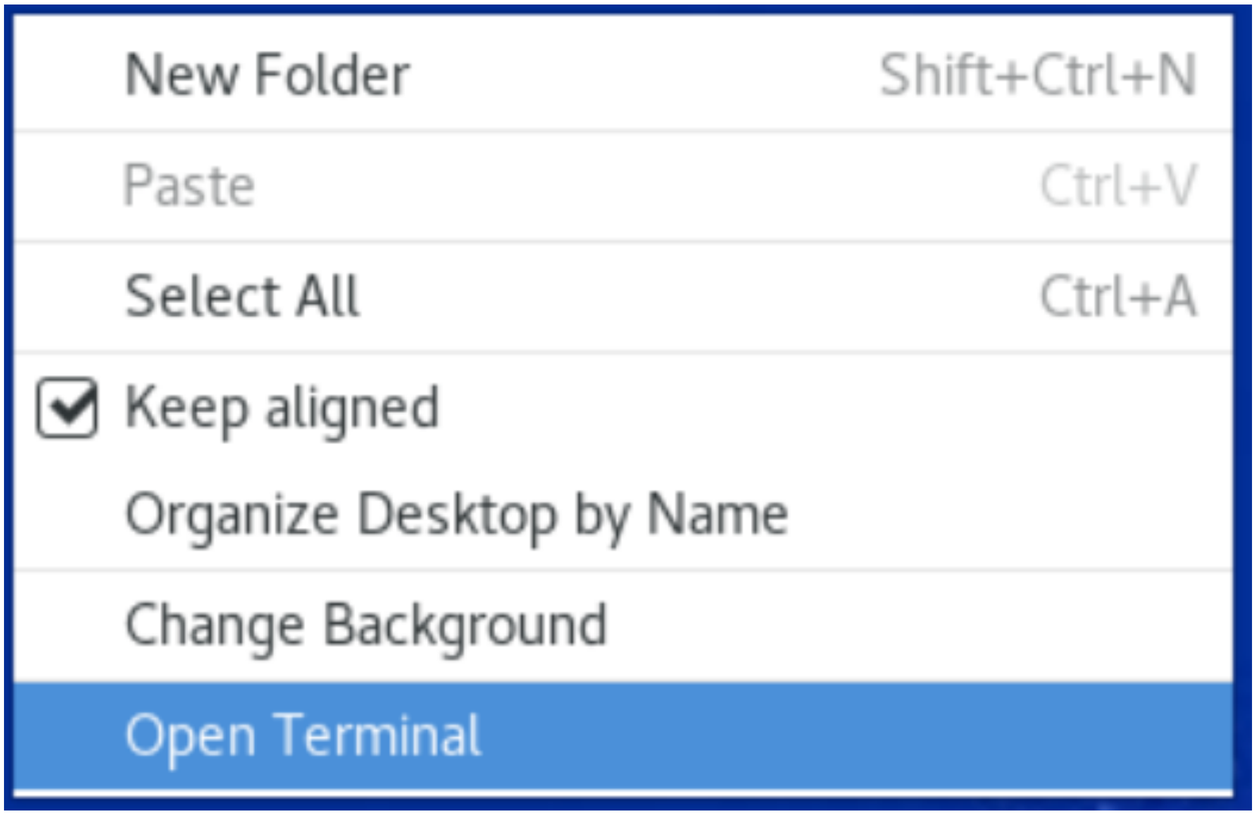
Java truststore
Use this for a Java client

PEM certificate
Use this for anything else

API endpoint

Applications and tools that use the REST producer API or the schema

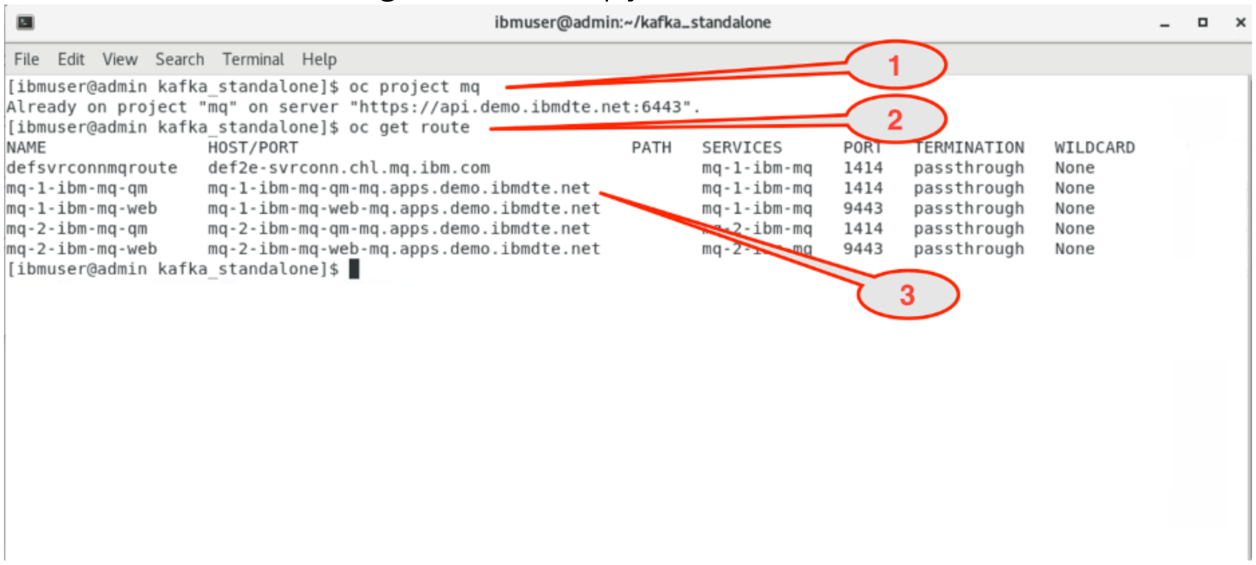
7. Many times during this lab, you need to open a terminal window. Click the right mouse on the workspace and click Open terminal.



8. Go to /home/ibmuser/Downloads directory: es-cert.jks and es-api-key.json. and copy or move both files to /home/ibmuser/esconfig.
   Within the /home/ibmuser/esconfig directory execute the following commands to move:
   mv /home/ibmuser/Downloads/es-cert.jks .
   mv /home/ibmuser/Downloads/es-api-key.json .
   Task 5 – Configuring Kafka MQ connectors (source and sink)
   Kafka connector can be implemented as: a docker application, Kubernetes or Stand-alone. Kafka Connect connectors that run inside a Java process called a worker. These instructions focus on stand-alone mode. You use a Kafka connector stand-alone JAR file (already installed for this lab).The connectors (sink and source) configuration file contain the properties that are needed for each connector. For this lab, you need to configure MQ Kafka Connectors.
   Source connector: https://github.com/ibm-messaging/kafka-connect-mq-source/tree/master/...

Sink connector: https://github.com/ibm-messaging/kafka-connect-mq-sink/tree/master/co...
The MQ source connector copies messages from a source MQ queue to a target Kafka topic. There is also an MQ sink connector that takes messages from a Kafka topic and transfers them to an MQ queue. You installed in desktop Machine a local Kafka. You do not need to update the connector config file (mq-source.properties and mq-sink.properties) for the value that is required to connect to your queue manager

1. To configure MQ connectors (source and sink), you need to get MQ hostname. Open a terminal window. Enter OpenShift command to work with MQ namespace: **oc project mq.** Enter the command to identify the MQ hostname, enter **oc get route**. Copy the address.

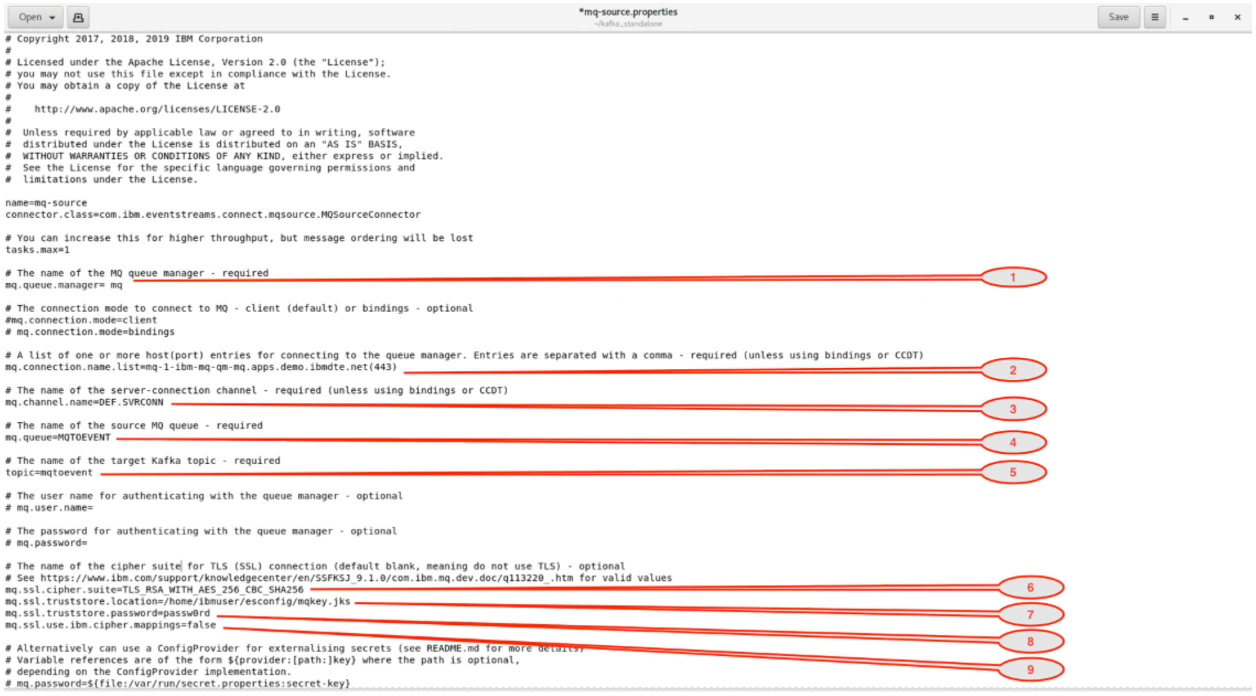

ibmuser@admin:~/kafka_standalone
File Edit View Search Terminal Help
1
2
3

```
[ibmuser@admin kafka_standalone]$ oc project mq
Already on project "mq" on server "https://api.demo.ibmdte.net:6443".
[ibmuser@admin kafka_standalone]$ oc get route
NAME                HOST/PORT                                   PATH   SERVICES     PORT   TERMINATION   WILDCARD
defsvrconnmqroute   def2e-svrconn.chl.mq.ibm.com                       mq-1-ibm-mq  1414   passthrough   None
mq-1-ibm-mq-qm      mq-1-ibm-mq-qm-mq.apps.demo.ibmdte.net             mq-1-ibm-mq  1414   passthrough   None
mq-1-ibm-mq-web     mq-1-ibm-mq-web-mq.apps.demo.ibmdte.net            mq-1-ibm-mq  9443   passthrough   None
mq-2-ibm-mq-qm      mq-2-ibm-mq-qm-mq.apps.demo.ibmdte.net             mq-2-ibm-mq  1414   passthrough   None
mq-2-ibm-mq-web     mq-2-ibm-mq-web-mq.apps.demo.ibmdte.net            mq-2-ibm-mq  9443   passthrough   None
[ibmuser@admin kafka_standalone]$ 
```

2. Configure **mq-source.properties** (*MQ to Event Streams*). Open a terminal window and go to /home/ibmuser/kafka_standalone/ directory and edit mq-source.properties. You can use **gedit mq-source.properties** and check the values as follows.

1. mq.queue.manager=**mq**
2. mq.connection.name.list= **mq-1-ibm-mq-qm-mq.apps.demo.ibmdte.net(443)**

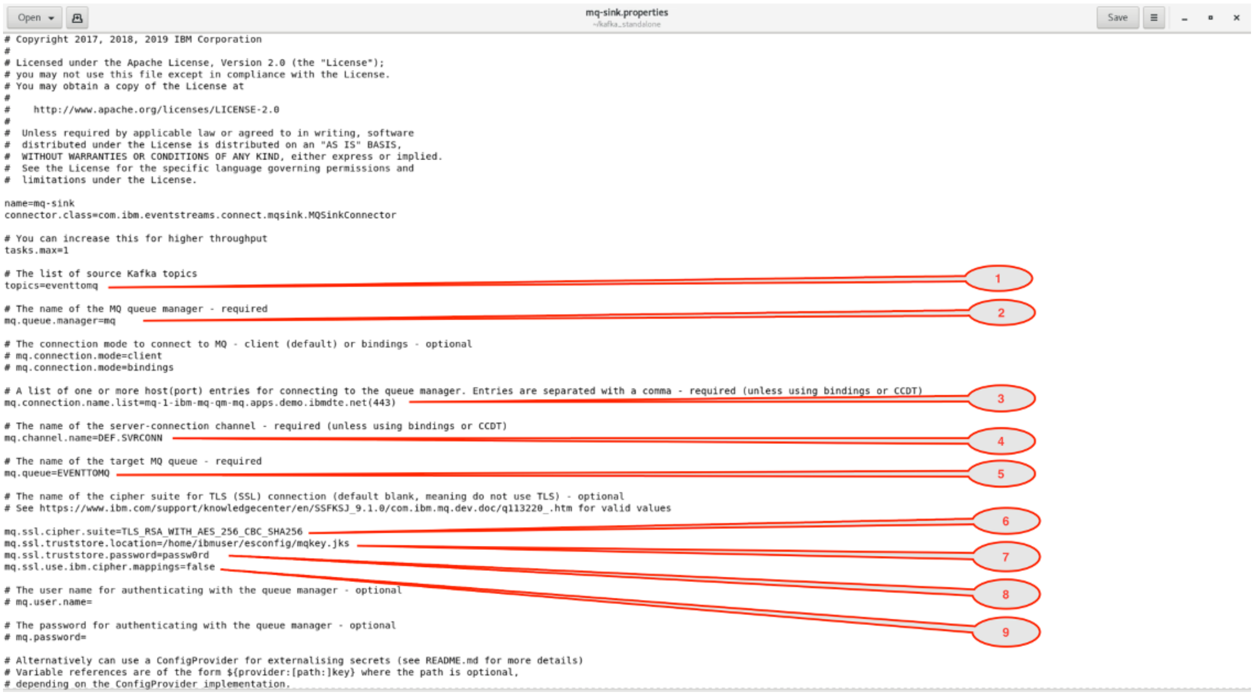
3. mq.channel.name=**DEF.SVRCONN**
4. mq.queue=**MQTOEVENT**
5. topic=**mqtoevent**
6. mq.ssl.cipher.suite**=TLS_RSA_WITH_AES_256_CBC_SHA256**
7. mq.ssl.truststore.location**=/home/ibmuser/esconfig/mqkey.jks** (we generated the key mqkey.jks – using keytool) – You can learn how to create this key, visit this link: https://developer.ibm.com/integration/blog/2020/02/28/connecting-to-a-q....
8. mq.ssl.truststore.password**=passw0rd**
9. mq.ssl.use.ibm.cipher.mappings**=false**
10. **Close** the source file.



```
# Copyright 2017, 2018, 2019 IBM Corporation
#
# Licensed under the Apache License, Version 2.0 (the "License");
# you may not use this file except in compliance with the License.
# You may obtain a copy of the License at
#
#    http://www.apache.org/licenses/LICENSE-2.0
#
#  Unless required by applicable law or agreed to in writing, software
#  distributed under the License is distributed on an "AS IS" BASIS,
#  WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
#  See the License for the specific language governing permissions and
#  limitations under the License.

name=mq-source
connector.class=com.ibm.eventstreams.connect.mqsource.MQSourceConnector

# You can increase this for higher throughput, but message ordering will be lost
tasks.max=1

# The name of the MQ queue manager - required
mq.queue.manager= mq

# The connection mode to connect to MQ - client (default) or bindings - optional
#mq.connection.mode=client
# mq.connection.mode=bindings

# A list of one or more host(port) entries for connecting to the queue manager. Entries are separated with a comma - required (unless using bindings or CCDT)
mq.connection.name.list=mq-1-ibm-mq-qm-mq.apps.demo.ibmdte.net(443)

# The name of the server-connection channel - required (unless using bindings or CCDT)
mq.channel.name=DEF.SVRCONN

# The name of the source MQ queue - required
mq.queue=MQTOEVENT

# The name of the target Kafka topic - required
topic=mqtoevent

# The user name for authenticating with the queue manager - optional
# mq.user.name=

# The password for authenticating with the queue manager - optional
# mq.password=

# The name of the cipher suite for TLS (SSL) connection (default blank, meaning do not use TLS) - optional
# See https://www.ibm.com/support/knowledgecenter/en/SSFKSJ_9.1.0/com.ibm.mq.dev.doc/q113220_.htm for valid values
mq.ssl.cipher.suite=TLS_RSA_WITH_AES_256_CBC_SHA256
mq.ssl.truststore.location=/home/ibmuser/esconfig/mqkey.jks
mq.ssl.truststore.password=passw0rd
mq.ssl.use.ibm.cipher.mappings=false

# Alternatively can use a ConfigProvider for externalising secrets (see README.md for more details)
# Variable references are of the form ${provider:[path:]key} where the path is optional,
# depending on the ConfigProvider implementation.
# mq.password=${file:/var/run/secret.properties:secret-key}
```

3. To configure mq-sink, use as the same you configure mq-source. Open a terminal window and go to /home/ibmuser/kafka_standalone/ directory and edit mq-source.properties, you can use **gedit mq-sink.properties** and check the values as follows:
1. topics=**eventtomq**
2. mq.queue.manager=**mq**

3. mq.connection.name.list= **mq-1-ibm-mq-qm-mq.apps.demo.ibmdte.net(443)**
4. mq.channel.name=**DEF.SVRCONN**
5. mq.queue=**EVENTTOMQ**
6. mq.ssl.cipher.suite=**TLS_RSA_WITH_AES_256_CBC_SHA256**
7. mq.ssl.truststore.location**=/home/ibmuser/esconfig/mqkey.jks**
8. mq.ssl.truststore.password=**passw0rd**
9. mq.ssl.use.ibm.cipher.mappings=**false**
10. **Close the sink file .**



```
# Copyright 2017, 2018, 2019 IBM Corporation
#
# Licensed under the Apache License, Version 2.0 (the "License");
# you may not use this file except in compliance with the License.
# You may obtain a copy of the License at
#
#    http://www.apache.org/licenses/LICENSE-2.0
#
#  Unless required by applicable law or agreed to in writing, software
#  distributed under the License is distributed on an "AS IS" BASIS,
#  WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
#  See the License for the specific language governing permissions and
#  limitations under the License.

name=mq-sink
connector.class=com.ibm.eventstreams.connect.mqsink.MQSinkConnector

# You can increase this for higher throughput
tasks.max=1

# The list of source Kafka topics
topics=eventtomq

# The name of the MQ queue manager - required
mq.queue.manager=mq

# The connection mode to connect to MQ - client (default) or bindings - optional
# mq.connection.mode=client
# mq.connection.mode=bindings

# A list of one or more host(port) entries for connecting to the queue manager. Entries are separated with a comma - required (unless using bindings or CCDT)
mq.connection.name.list=mq-1-ibm-mq-qm-mq.apps.demo.ibmdte.net(443)

# The name of the server-connection channel - required (unless using bindings or CCDT)
mq.channel.name=DEF.SVRCONN

# The name of the target MQ queue - required
mq.queue=EVENTTOMQ

# The name of the cipher suite for TLS (SSL) connection (default blank, meaning do not use TLS) - optional
# See https://www.ibm.com/support/knowledgecenter/en/SSFKSJ_9.1.0/com.ibm.mq.dev.doc/q113220_.htm for valid values

mq.ssl.cipher.suite=TLS_RSA_WITH_AES_256_CBC_SHA256
mq.ssl.truststore.location=/home/ibmuser/esconfig/mqkey.jks
mq.ssl.truststore.password=passw0rd
mq.ssl.use.ibm.cipher.mappings=false

# The user name for authenticating with the queue manager - optional
# mq.user.name=

# The password for authenticating with the queue manager - optional
# mq.password=

# Alternatively can use a ConfigProvider for externalising secrets (see README.md for more details)
# Variable references are of the form ${provider:[path:]key} where the path is optional,
# depending on the ConfigProvider implementation.
```

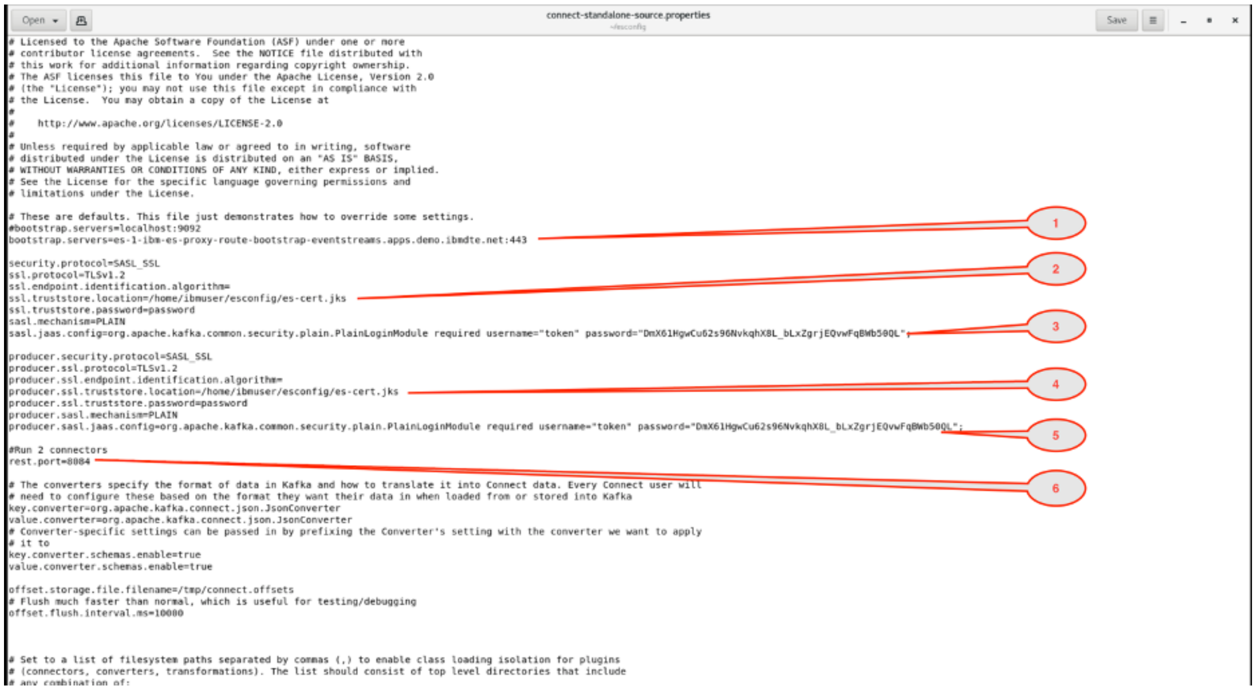
Task 6 – Configuring the Connectors
Download Kafka.
You need the Kafka environment to run the MQ connectors in stand-alone mode. This can be downloaded from the Kafka project, but it has already been downloaded onto this image. It is stored in /home/ibmuser/esconfig/kafka_2.13-2.5.0.tgz.
Copy kafka_2.13-2.5.0.tgz to /home/ibmuser directory and enter **tar -xvf kafka_2.13-2.5.0.tgz**. The installation creates several directories including bin for the Kafka executables and config for the configuration

files. You can download the latest version of Kafka server from this download: http://kafka.apache.org/downloads
Note:
Within the /home/ibmuser/ directory execute the following commands:
mv /home/ibmuser/esconfig/kafka_2.13-2.5.0.tgz

1. Make sure that **es-cert.jks** and **es-api-key.json** are in /home/ibmuser/esconfig directory
2. Configure **connect-standalone-source.properties**. From the terminal window, edit (use gedit) /home/ibmuser/esconfig/**connect-standalone-source.properties**.There are two sessions that are necessary to configure in **security** and **producer.**
3. The first session is to configure **security parameters** (Verify):

1. bootstrap.servers: **es-1-ibm-es-proxy-route-bootstrap-eventstreams.apps.demo.ibmdte.net:443** (You saved in a file as **cred.txt**)
2. Verify the **ssl.truststore.location=/home/ibmuser/esconfig/es-cert.jks**
3. You need to type **API Key password** (edit /home/ibmuser/esconfig/es-api-key.json and copy the API Key). Type or Paste as password the API Key as sasl.jaas.config=org.apache.kafka.common.security.plain.PlainLoginModule required username="token" password="**DmX61HgwCu62s96NvkqhX8L_bLxZgrjEQvwFqBWb50QL**";
   The second session is to configure **producer parameters** (Verify):

4. Verify the producer.ssl.truststore.location**=/home/ibmuser/esconfig/es-cert.jks**Paste as password the API Key as producer.sasl.jaas.config=org.apache.kafka.common.security.plain.PlainLoginModule required username="token" password=**" DmX61HgwCu62s96NvkqhX8L_bLxZgrjEQvwFqBWb50QL** " .
5. As you run two connectors source and sink, it is necessary to change rest.port to **8084.**
6. Click **Save** to save the file and then **close** it.



```
# Licensed to the Apache Software Foundation (ASF) under one or more
# contributor license agreements.  See the NOTICE file distributed with
# this work for additional information regarding copyright ownership.
# The ASF licenses this file to You under the Apache License, Version 2.0
# (the "License"); you may not use this file except in compliance with
# the License.  You may obtain a copy of the License at
#
#    http://www.apache.org/licenses/LICENSE-2.0
#
# Unless required by applicable law or agreed to in writing, software
# distributed under the License is distributed on an "AS IS" BASIS,
# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
# See the License for the specific language governing permissions and
# limitations under the License.

# These are defaults. This file just demonstrates how to override some settings.
#bootstrap.servers=localhost:9092
bootstrap.servers=es-1-ibm-es-proxy-route-bootstrap-eventstreams.apps.demo.ibmdte.net:443

security.protocol=SASL_SSL
ssl.protocol=TLSv1.2
ssl.endpoint.identification.algorithm=
ssl.truststore.location=/home/ibmuser/esconfig/es-cert.jks
ssl.truststore.password=password
sasl.mechanism=PLAIN
sasl.jaas.config=org.apache.kafka.common.security.plain.PlainLoginModule required username="token" password="DmX61HgwCu62s96NvkqhX8L_bLxZgrjEQvwFqBWb50QL";

producer.security.protocol=SASL_SSL
producer.ssl.protocol=TLSv1.2
producer.ssl.endpoint.identification.algorithm=
producer.ssl.truststore.location=/home/ibmuser/esconfig/es-cert.jks
producer.ssl.truststore.password=password
producer.sasl.mechanism=PLAIN
producer.sasl.jaas.config=org.apache.kafka.common.security.plain.PlainLoginModule required username="token" password="DmX61HgwCu62s96NvkqhX8L_bLxZgrjEQvwFqBWb50QL";

#Run 2 connectors
rest.port=8084

# The converters specify the format of data in Kafka and how to translate it into Connect data. Every Connect user will
# need to configure these based on the format they want their data in when loaded from or stored into Kafka
key.converter=org.apache.kafka.connect.json.JsonConverter
value.converter=org.apache.kafka.connect.json.JsonConverter
# Converter-specific settings can be passed in by prefixing the Converter's setting with the converter we want to apply
# it to
key.converter.schemas.enable=true
value.converter.schemas.enable=true

offset.storage.file.filename=/tmp/connect.offsets
# Flush much faster than normal, which is useful for testing/debugging
offset.flush.interval.ms=10000



# Set to a list of filesystem paths separated by commas (,) to enable class loading isolation for plugins
# (connectors, converters, transformations). The list should consist of top level directories that include
# any combination of:
```

4. Configure connect-standalone-sink.properties : (similar as **producer configuration**).
5. There are two sessions that are necessary to configure in **security** and **producer.**
6. The first session that you configure is the **security**:
1. bootstrap.servers: **es-1-ibm-es-proxy-route-bootstrap-eventstreams.apps.demo.ibmdte.net:443** (You saved in a file as **cred.txt**).
2. Verify the **ssl.truststore.location=/home/ibmuser/esconfig/es-cert.jks**
3. Paste as password the API Key as sasl.jaas.config=org.apache.kafka.common.security.plain.PlainLoginModule required username="token" password="**DmX61HgwCu62s96NvkqhX8L_bLxZgrjEQvwFqBWb50QL**";

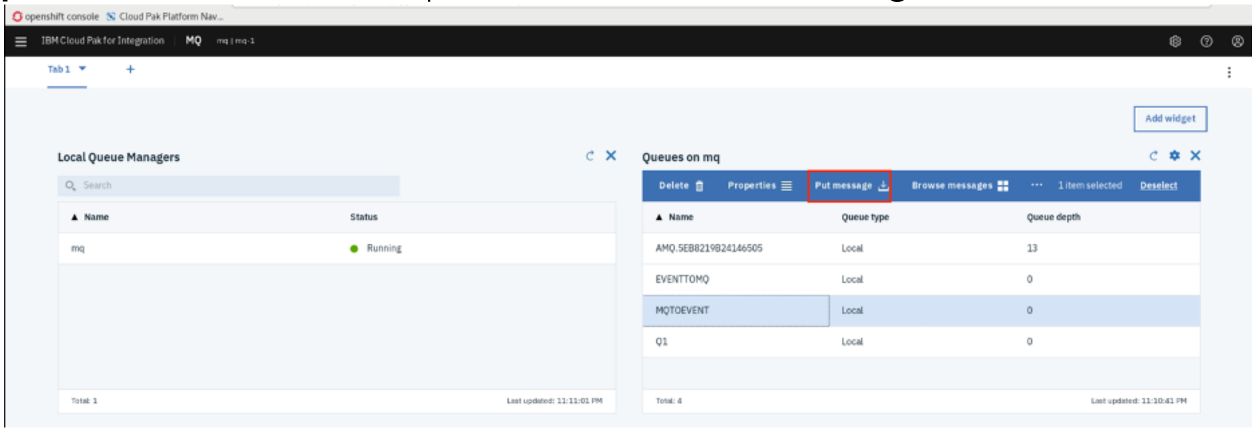
   The second session is to configure **consumer parameters** (Verify or enter):

4. Verify the producer.ssl.truststore.location**=/home/ibmuser/esconfig/es-cert.jks**
5. Type or Paste as password the API Key as consumer.sasl.jaas.config=org.apache.kafka.common.security.plain.PlainLoginModule

required username="token" password="**
DmX61HgwCu62s96NvkqhX8L_bLxZgrjEQvwFqBWb50QL** "
6. Click **Save** to save the file and then **close** it

Task 7 – Executing and Testing MQ Connectors
To use an existing Kafka cluster, specify the connection information in the connector configuration file.

1. Open a terminal window.
1. Copy /home/ibmuser/esconfig/connect-standalone-source.properties to /home/ibmuser/kafka_2.13-2.5.0/config.
2. Copy /home/ibmuser/esconfig/connect-standalone-sink.properties to /home/ibmuser/kafka_2.13-2.5.0/config.
3. Copy two shell scripts: event2mq.sh and mq2event.sh from /home/ibmuser/esconfig/ to /home/ibmuser/kafka_2.13-2.5.0/ .
4. To start the source connector in the terminal window and go to /home/ibmuser/kafka_2.13-2.5.0 and enter **./mq2event.sh** or **CLASSPATH=/home/ibmuser/kafka-connect-mq-source-1.3.0-jar-with-dependencies.jar bin/connect-standalone.sh config/connect-standalone-source.properties /home/ibmuser/kafka_standalone/mq-source.properties** .
5. Check the log.
2. Go to the **mq-1** instance and open MQ console, locate **Queues on mq.** select **MQTOEVENT** queue and click **Put message.**



Queues on mq

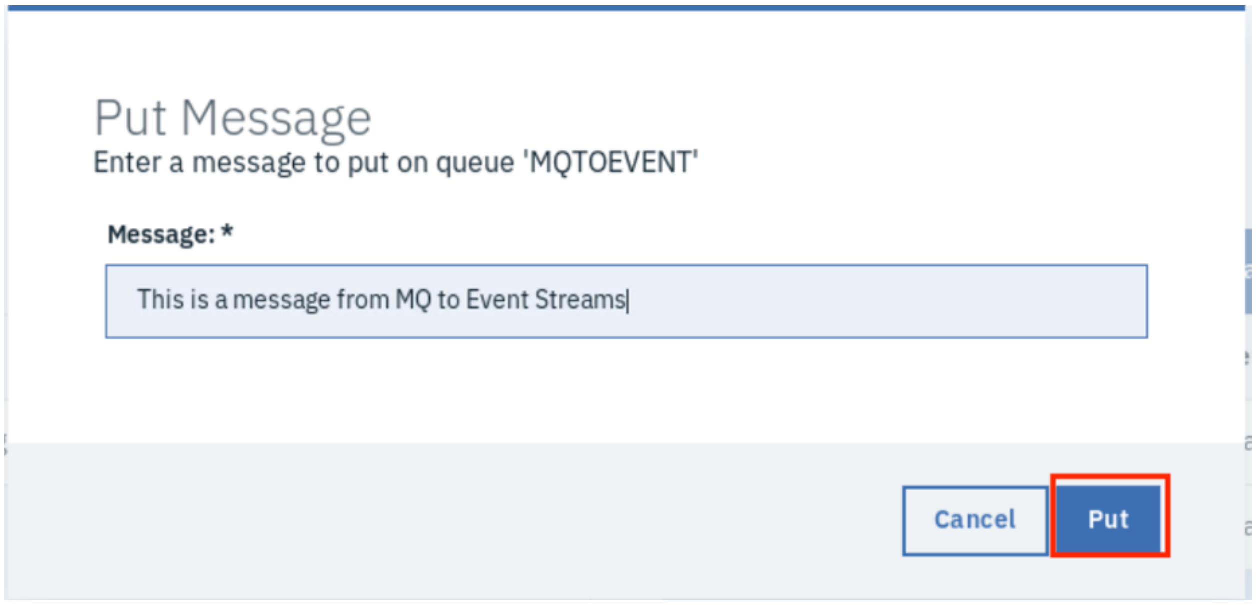
Name	Status
mq	● Running

Total: 1 Last updated: 11:11:01 PM

Name	Queue type	Queue depth
AMQ.5EB8219B24146505	Local	13
EVENTTOMQ	Local	0
MQTOEVENT	Local	0
Q1	Local	0

Total: 4 Last updated: 11:10:41 PM

3. Type a message (***suggestion***: **This is a message from MQ to Event Streams**) on **Message** and click **Put.** Notice that there is no message on the **MQTOEVENT** queue. As soon as you put a message, immediately the connector sends it to Event Streams.



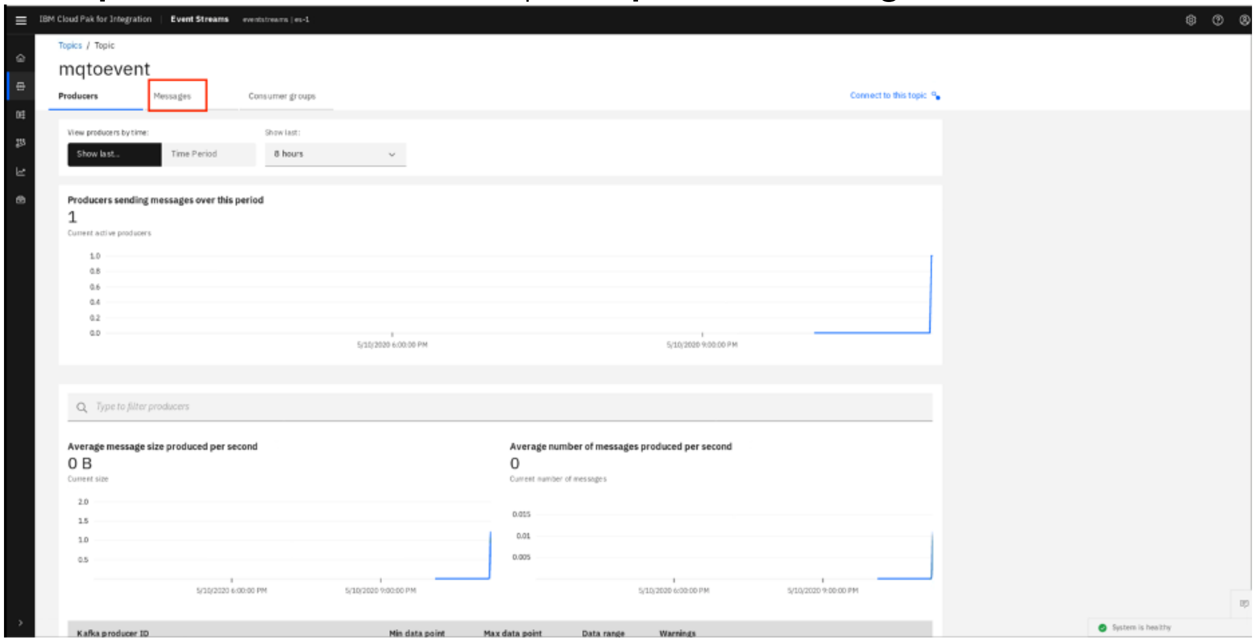
Put Message

Enter a message to put on queue 'MQTOEVENT'

Message: *

This is a message from MQ to Event Streams

4. Return to the Event Streams main page and click **Topics**. Click the topic **mqtovent** and to see the topic **mqtovent messages**.



mqtoevent

View producers by time:

Show last: 8 hours

Producers sending messages over this period

1

Current active producers

Average message size produced per second

0 B

Current size

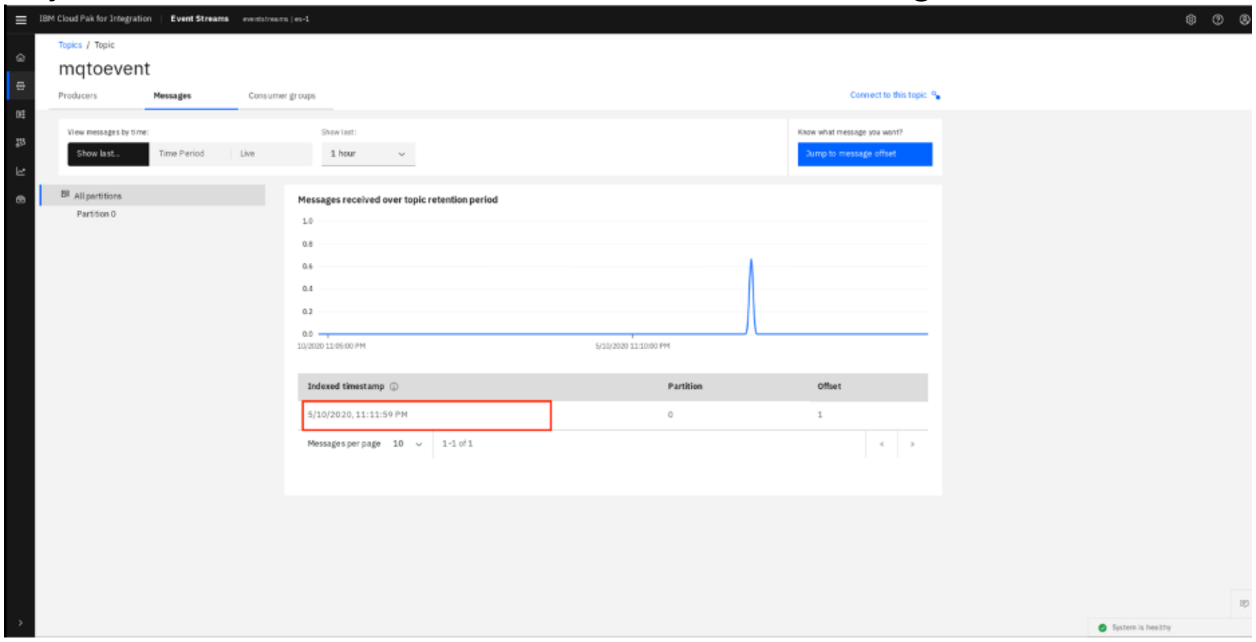
Average number of messages produced per second

0

Current number of messages

Kafka producer ID	Min data point	Max data point	Data range	Warnings

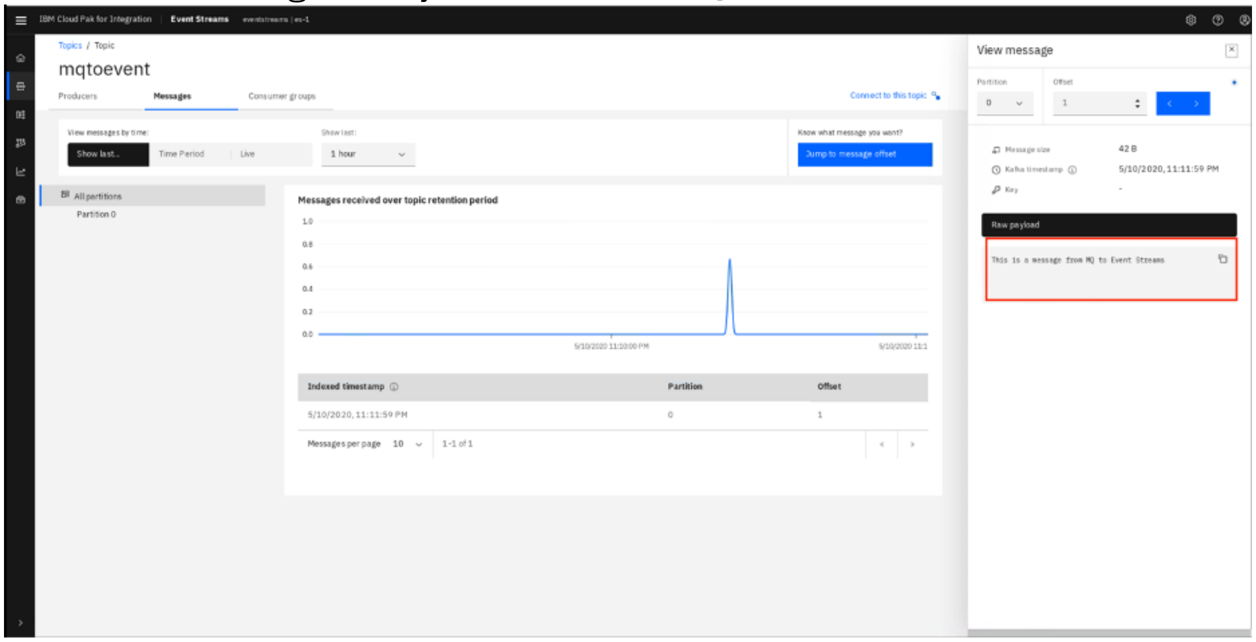
5. You can view the time frame of data to display (Hours). You can change it to Days, Hours, Minutes and Seconds. Click the **message**.



Indexed timestamp	Partition	Offset
5/10/2020, 11:11:59 PM	0	1

Messages per page 10 1-1 of 1

6. You see the message that you sent from MQ to Event Streams.



Indexed timestamp	Partition	Offset
5/10/2020, 11:11:59 PM	0	1

Messages per page 10 1-1 of 1

Partition	0
Offset	1
Message size	42 B
Kafka timestamp	5/10/2020, 11:11:59 PM
Key	-

Raw payload

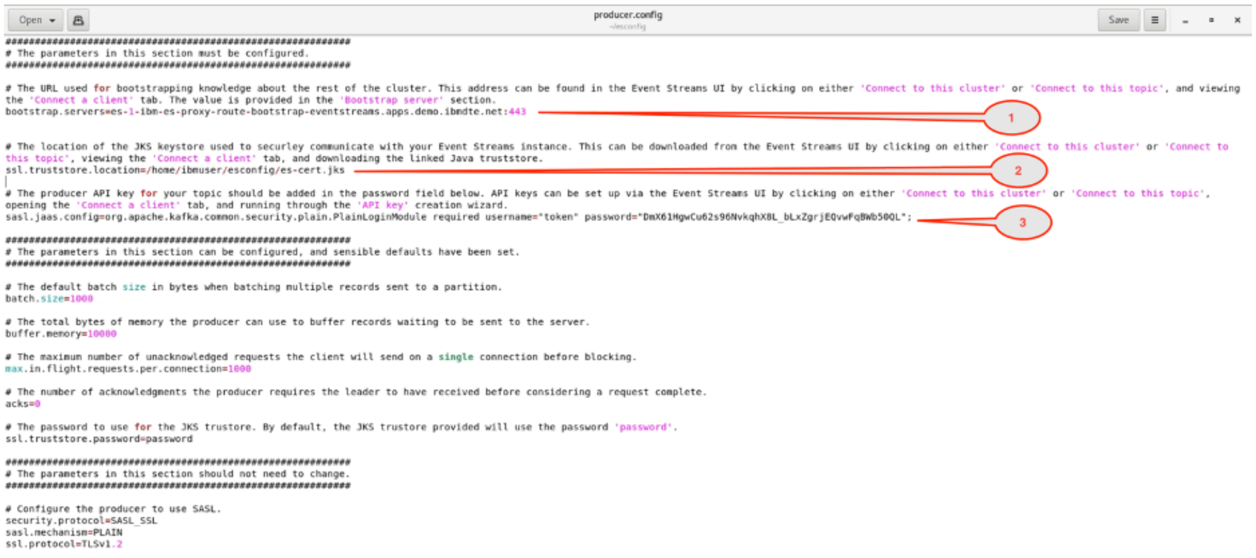
```
This is a message from MQ to Event Streams
```

7. Keep the source connector running. Open a new terminal window and go to /home/ibmuser/kafka_2.11-2.5.0 and type: **./event2mq.sh or CLASSPATH=/home/ibmuser/kafka-connect-mq-sink-1.3.0-jar-with-**

dependencies.jar bin/connect-standalone.sh config/connect-standalone-sink.properties /home/ibmuser/kafka_standalone/mq-sink.properties .

1. Check the log.
8. In order to send events, use a file (you can find it on /home/ibmuser/esconfig/file_example_XLS_50.csv) to simulate events from Event Streams to MQ.
9. To send the lines of this file, you use **producer.jar** . Open a terminal window and go to /home/ibmuser/esconfig directory and edit producer.config (use **gedit producer.config**) and enter or verify the parameters.

1. Verify Bootstrap server: **es-1-ibm-es-proxy-route-bootstrap-eventstreams.apps.demo.ibmdte.net:443**
2. Verify ssl.truststore.location: **/home/ibmuser/esconfig/es-cert.jks**
3. Enter password (It is the es-api-key.json that you use in connect-standalone-sink-properties)**.** sasl.jaas.config=org.apache.kafka.common.security.plain.PlainLoginModule required username="token" password="**DmX61HgwCu62s96NvkqhX8L_bLxZgrjEQvwFqBWb50QL**"
4. Save and close the file.

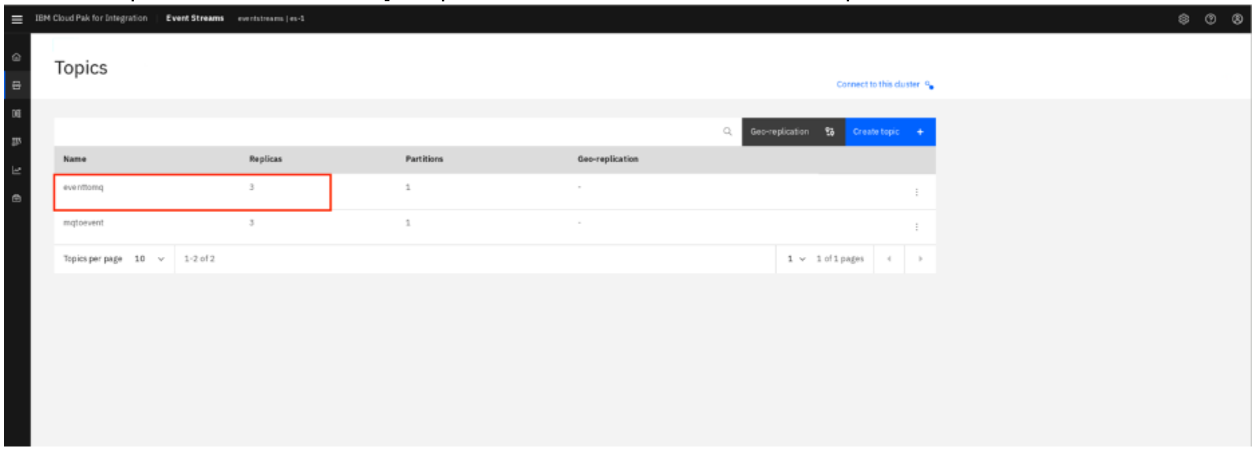


```
################################################################
# The parameters in this section must be configured.
################################################################

# The URL used for bootstrapping knowledge about the rest of the cluster. This address can be found in the Event Streams UI by clicking on either 'Connect to this cluster' or 'Connect to this topic', and viewing the 'Connect a client' tab. The value is provided in the 'Bootstrap server' section.
bootstrap.servers=es-1-ibm-es-proxy-route-bootstrap-eventstreams.apps.demo.ibmdte.net:443

# The location of the JKS keystore used to securley communicate with your Event Streams instance. This can be downloaded from the Event Streams UI by clicking on either 'Connect to this cluster' or 'Connect to this topic', viewing the 'Connect a client' tab, and downloading the linked Java truststore.
ssl.truststore.location=/home/ibmuser/esconfig/es-cert.jks

# The producer API key for your topic should be added in the password field below. API keys can be set up via the Event Streams UI by clicking on either 'Connect to this cluster' or 'Connect to this topic', opening the 'Connect a client' tab, and running through the 'API key' creation wizard.
sasl.jaas.config=org.apache.kafka.common.security.plain.PlainLoginModule required username="token" password="DmX61HgwCu62s96NvkqhX8L_bLxZgrjEQvwFqBWb50QL";

################################################################
# The parameters in this section can be configured, and sensible defaults have been set.
################################################################

# The default batch size in bytes when batching multiple records sent to a partition.
batch.size=1000

# The total bytes of memory the producer can use to buffer records waiting to be sent to the server.
buffer.memory=10000

# The maximum number of unacknowledged requests the client will send on a single connection before blocking.
max.in.flight.requests.per.connection=1000

# The number of acknowledgments the producer requires the leader to have received before considering a request complete.
acks=0

# The password to use for the JKS trustore. By default, the JKS trustore provided will use the password 'password'.
ssl.truststore.password=password

################################################################
# The parameters in this section should not need to change.
################################################################

# Configure the producer to use SASL.
security.protocol=SASL_SSL
sasl.mechanism=PLAIN
ssl.protocol=TLSv1.2
```

Now, you send messages to MQ. Use a Java application(es-producer.jar) to produce Events from Event Streams to MQ (you send

50 events).

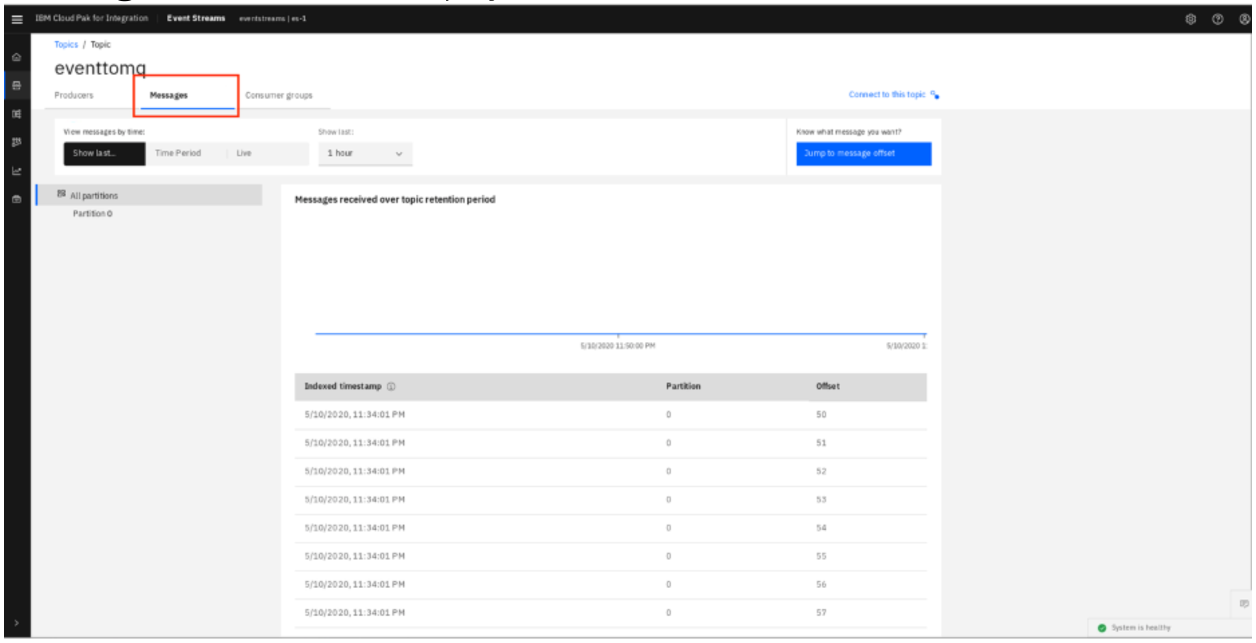
10. In this terminal go to directory /home/ibmuser/esconfig/ and type ./loadevent.sh or **java -jar es-producer.jar -t eventtomq -T 1000 -n 50 -f /home/ibmuser/esconfig/file_example_XLS_50.csv**. Have a look at the log and see the results:
11. Open **eventtomq** topic in Event Streams topics.



Name	Replicas	Partitions	Geo-replication
eventtomq	3	1	-
mqtoevent	3	1	-

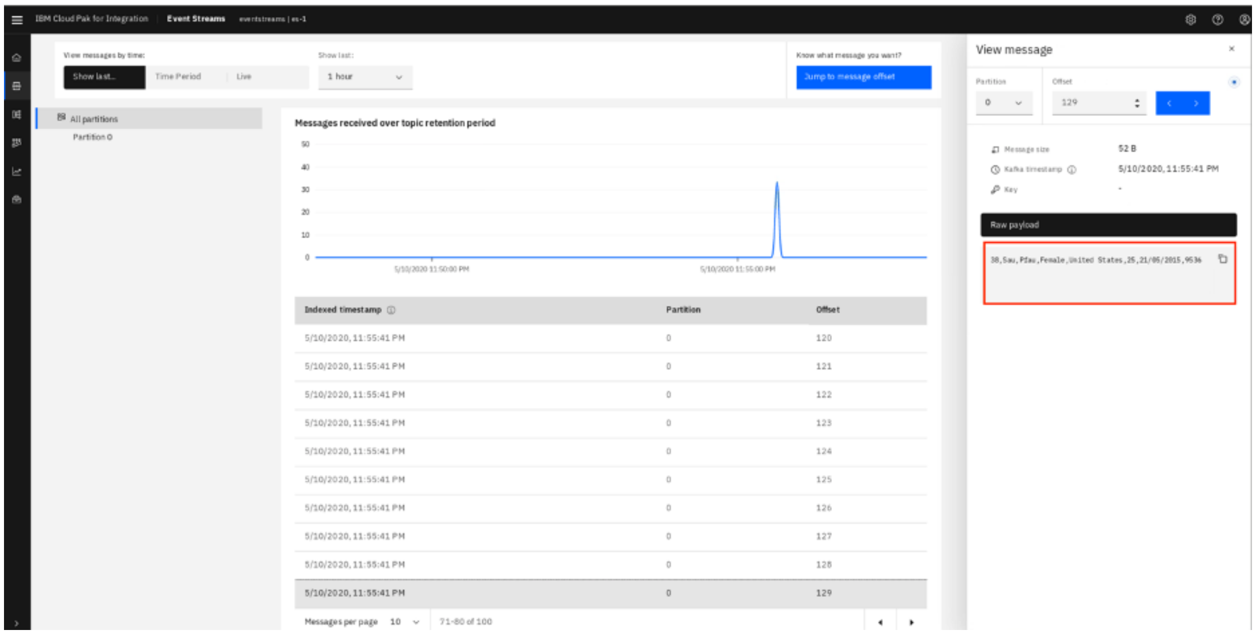
Topics per page 10 1-2 of 2 1 1 of 1 pages

12. Open **Message** and see the list of events that you sent. Click one of messages and check the payload.



Indexed timestamp	Partition	Offset
5/10/2020, 11:34:01 PM	0	50
5/10/2020, 11:34:01 PM	0	51
5/10/2020, 11:34:01 PM	0	52
5/10/2020, 11:34:01 PM	0	53
5/10/2020, 11:34:01 PM	0	54
5/10/2020, 11:34:01 PM	0	55
5/10/2020, 11:34:01 PM	0	56
5/10/2020, 11:34:01 PM	0	57

13. Select one message and see the message.


IBM Cloud Pak for Integration | Event Streams eventstreams | es-1

View messages by time: Show last... | Time Period | Live

Show last: 1 hour

Know what message you want? Jump to message offset

All partitions

Partition 0

Messages received over topic retention period

Indexed timestamp	Partition	Offset
5/10/2020, 11:55:41 PM	0	120
5/10/2020, 11:55:41 PM	0	121
5/10/2020, 11:55:41 PM	0	122
5/10/2020, 11:55:41 PM	0	123
5/10/2020, 11:55:41 PM	0	124
5/10/2020, 11:55:41 PM	0	125
5/10/2020, 11:55:41 PM	0	126
5/10/2020, 11:55:41 PM	0	127
5/10/2020, 11:55:41 PM	0	128
5/10/2020, 11:55:41 PM	0	129

Messages per page 10 | 71–80 of 100

View message

Partition: 0 | Offset: 129

Message size: 52 B

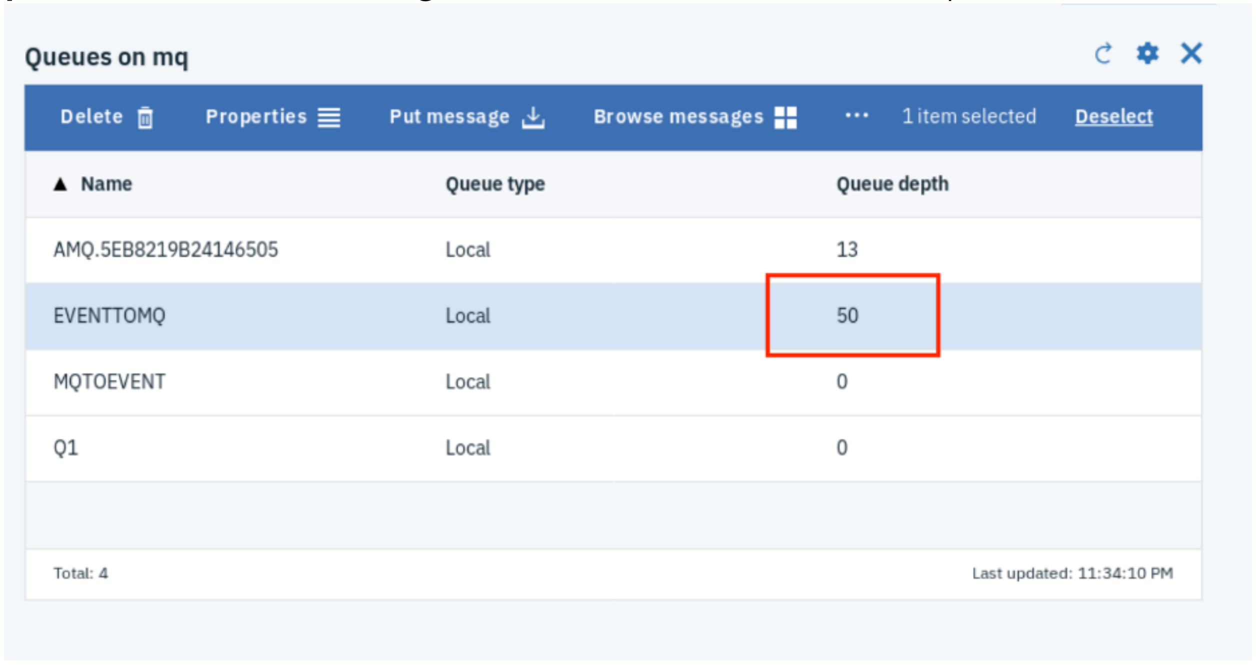
Kafka timestamp: 5/10/2020, 11:55:41 PM

Key: -

Raw payload

```
38,Sau,Pfau,Female,United States,25,21/05/2015,9536
```

14. Go to MQ Console, click the Refresh icon and check the **Queue Depth. You see 50 messages arrived in EVENTTOMQ** queue**.**

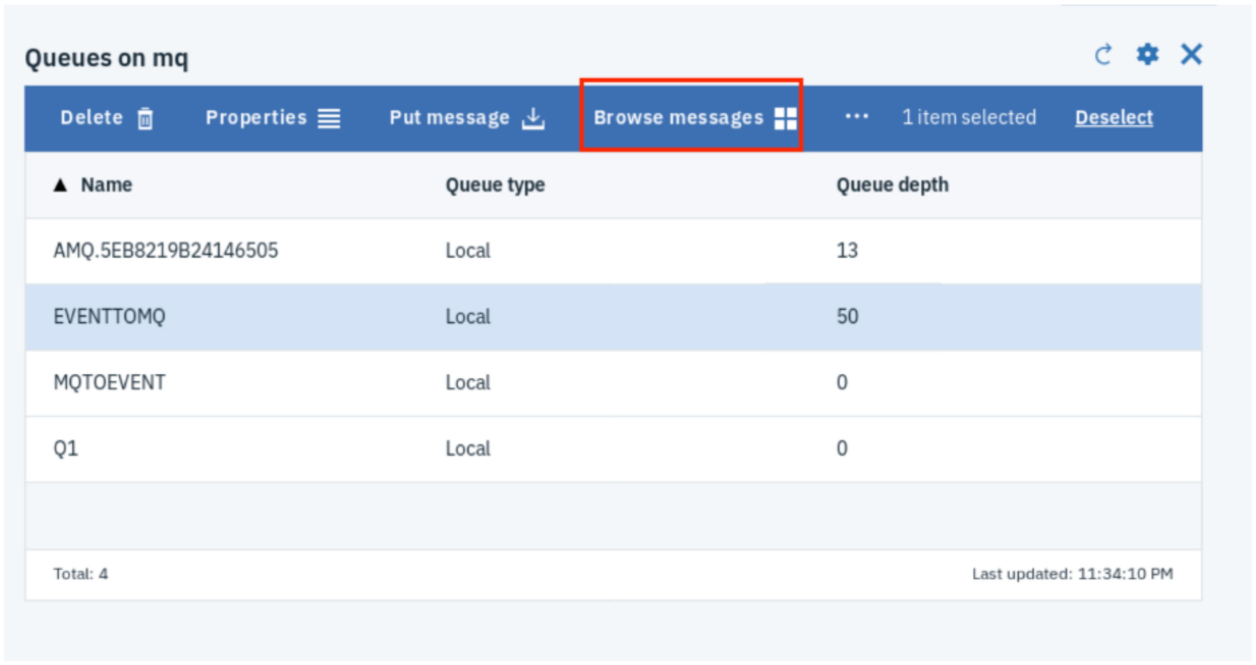

Queues on mq

Delete | Properties | Put message | Browse messages | ... | 1 item selected | Deselect

▲ Name	Queue type	Queue depth
AMQ.5EB8219B24146505	Local	13
EVENTTOMQ	Local	50
MQTOEVENT	Local	0
Q1	Local	0

Total: 4 | Last updated: 11:34:10 PM

15. Select **EVENTTOMQ** and click **Browse messages**.

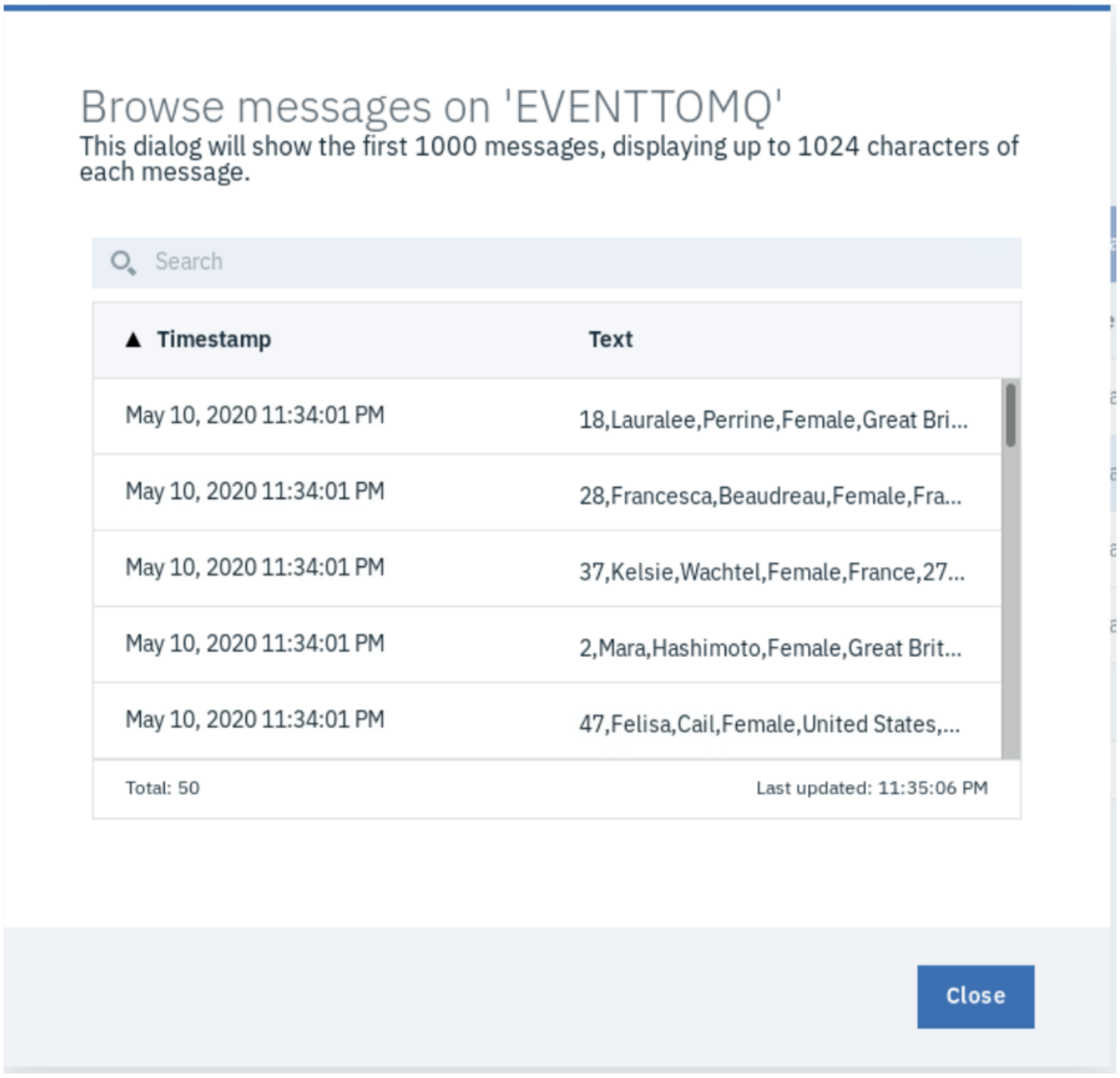


▲ Name	Queue type	Queue depth
AMQ.5EB8219B24146505	Local	13
EVENTTOMQ	Local	50
MQTOEVENT	Local	0
Q1	Local	0

Total: 4

Last updated: 11:34:10 PM

16. A pop-up window check the messages from Event Streams.



Browse messages on 'EVENTTOMQ'

This dialog will show the first 1000 messages, displaying up to 1024 characters of each message.

▲ Timestamp	Text
May 10, 2020 11:34:01 PM	18,Lauralee,Perrine,Female,Great Bri...
May 10, 2020 11:34:01 PM	28,Francesca,Beaudreau,Female,Fra...
May 10, 2020 11:34:01 PM	37,Kelsie,Wachtel,Female,France,27...
May 10, 2020 11:34:01 PM	2,Mara,Hashimoto,Female,Great Brit...
May 10, 2020 11:34:01 PM	47,Felisa,Cail,Female,United States,...

Total: 50 Last updated: 11:35:06 PM

Summary

You've successfully completed the tutorial. You were able to add a layer of secure, reliable, event-driven, and real time data which can be re-used across applications in your enterprise. You learned how to:

- Configure message queues
- Create event streams topics
- Configure message queue connectors (sink and source)
- Execute a test run of the flow and view the data

To try out more labs, go to Cloud Pak for Integration Demos. For more information about the Cloud Pak for Integration, go to https://www.ibm.com/cloud/cloud-pak-for-integration.