Red Hat AMQ 2020.Q4

Evaluating AMQ Streams on OpenShift

For use with AMQ Streams 1.6 on OpenShift Container Platform

Last Updated: 2020-12-02

Red Hat AMQ 2020.Q4 Evaluating AMQ Streams on OpenShift

For use with AMQ Streams 1.6 on OpenShift Container Platform

Legal Notice

Abstract

This guide describes how to install and manage AMQ Streams to evaluate its potential use in a production environment.

Table of Contents

CHAPTER 1. OVERVIEW OF AMQ STREAMS

AMQ Streams simplifies the process of running Apache Kafka in an OpenShift cluster.

This guide provides instructions for evaluating a working environment of AMQ Streams. The steps describe how to get a AMQ Streams deployment up-and-running as quickly as possible.

Before trying AMQ Streams, it is useful to understand its capabilities and how you might wish to use it. This chapter introduces some of the key concepts behind Kafka, and also provides a brief overview of the AMQ Streams Operators.

Operators are a method of packaging, deploying, and managing an OpenShift application. AMQ Streams Operators extend OpenShift functionality, automating common and complex tasks related to a Kafka deployment. By implementing knowledge of Kafka operations in code, Kafka administration tasks are simplified and require less manual intervention.

1.1. KAFKA CAPABILITIES

The underlying data stream-processing capabilities and component architecture of Kafka can deliver:

- Microservices and other applications to share data with extremely high throughput and low latency
- Message ordering guarantees
- Message rewind/replay from data storage to reconstruct an application state
- Message compaction to remove old records when using a key-value log
- Horizontal scalability in a cluster configuration
- Replication of data to control fault tolerance
- Retention of high volumes of data for immediate access

1.2. KAFKA USE CASES

Kafka's capabilities make it suitable for:

- Event-driven architectures
- Event sourcing to capture changes to the state of an application as a log of events
- Message brokering
- Website activity tracking
- Operational monitoring through metrics
- Log collection and aggregation
- Commit logs for distributed systems
- Stream processing so that applications can respond to data in real time

1.3. HOW AMQ STREAMS SUPPORTS KAFKA

AMQ Streams provides container images and Operators for running Kafka on OpenShift. AMQ Streams Operators are fundamental to the running of AMQ Streams. The Operators provided with AMQ Streams are purpose-built with specialist operational knowledge to effectively manage Kafka.

Operators simplify the process of:

- Deploying and running Kafka clusters
- Deploying and running Kafka components
- Configuring access to Kafka
- Securing access to Kafka
- Upgrading Kafka
- Managing brokers
- Creating and managing topics
- Creating and managing users

1.4. OPERATORS

AMQ Streams provides Operators for managing a Kafka cluster running within an OpenShift cluster.

Cluster Operator

Deploys and manages Apache Kafka clusters, Kafka Connect, Kafka MirrorMaker, Kafka Bridge, Kafka Exporter, and the Entity Operator

Entity Operator

Comprises the Topic Operator and User Operator

Topic Operator

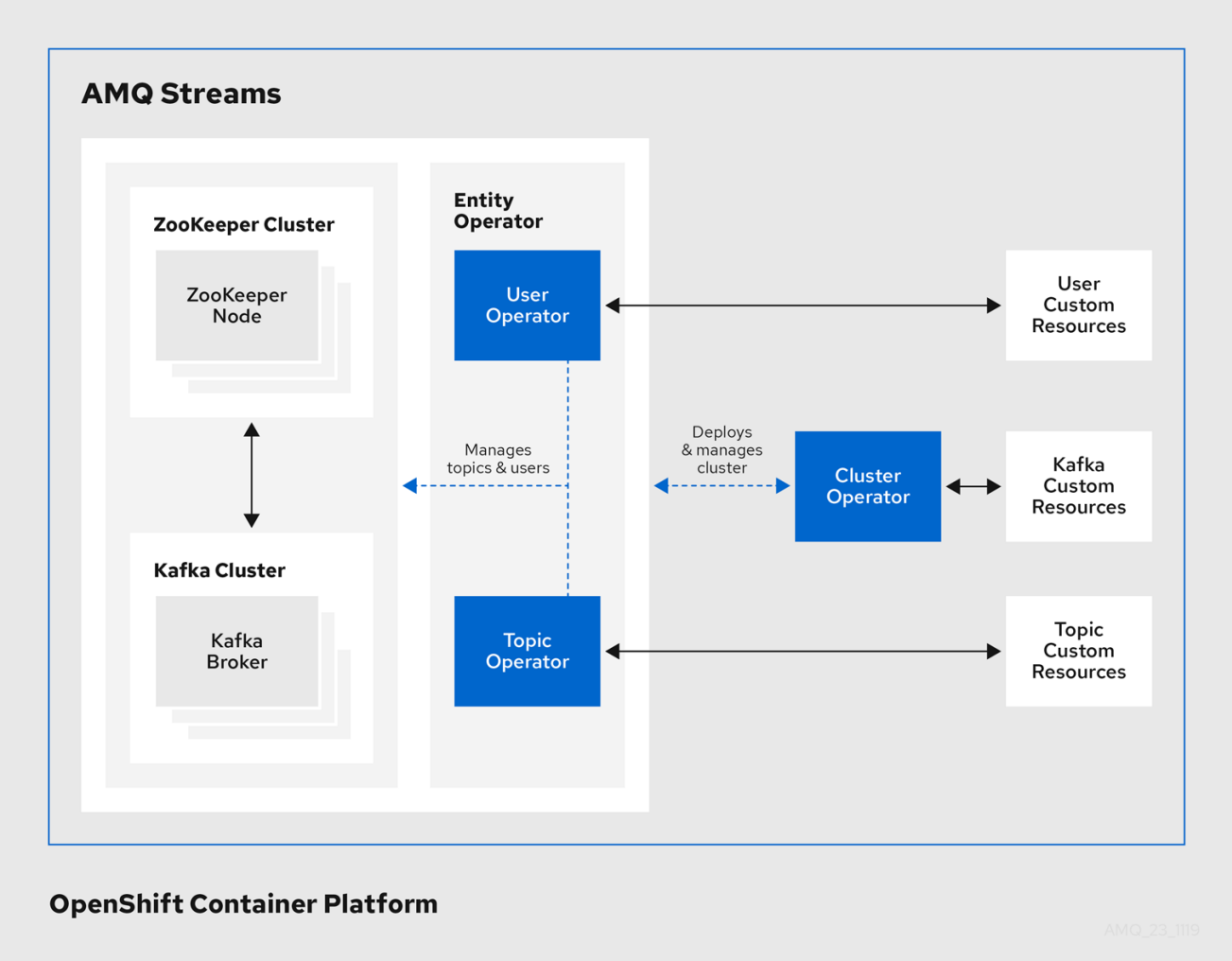
Manages Kafka topics

User Operator

Manages Kafka users

The Cluster Operator can deploy the Topic Operator and User Operator as part of an **Entity Operator** configuration at the same time as a Kafka cluster.

Operators within the AMQ Streams architecture



1.5. DOCUMENT CONVENTIONS

Replaceables

In this document, replaceable text is styled in **monospace**, with italics, uppercase, and hyphens.

For example, in the following code, you will want to replace ***MY-NAMESPACE*** with the name of your namespace:

```
sed -i 's/namespace: .*/namespace: MY-NAMESPACE/' install/cluster-operator/*RoleBinding*.yaml
```

CHAPTER 2. EVALUATE AMQ STREAMS

The procedures in this chapter provide a quick way to evaluate the functionality of AMQ Streams.

Follow the steps in the order provided to install AMQ Streams, and start sending and receiving messages from a topic:

- Ensure you have the required prerequisites
- Install AMQ Streams
- Create a Kafka cluster
- Enable authentication for secure access to the Kafka cluster
- Access the Kafka cluster to send and receive messages

Ensure you have the prerequisites and then follow the tasks in the order provided in this chapter.

2.1. PREREQUISITES

- An OpenShift Container Platform cluster (3.11 and later) running on which to deploy AMQ Streams must be running.
- You need to be able to access the AMQ Streams download site .

2.2. DOWNLOADING AMQ STREAMS

A ZIP file contains the resources required for installation of AMQ Streams, along with examples for configuration.

Procedure

1. Ensure your subscription has been activated and your system is registered.
   For more information about using the Customer Portal to activate your Red Hat subscription and register your system for packages, see Appendix A, *Using Your Subscription*.
2. Download the **amq-streams-x.y.z-ocp-install-examples.zip** file from the AMQ Streams download site.
3. Unzip the file to any destination.
   - Windows or Mac: Extract the contents of the ZIP archive by double clicking on the ZIP file.
   - Red Hat Enterprise Linux: Open a terminal window in the target machine and navigate to where the ZIP file was downloaded.

   Extract the ZIP file with this command:

   ```
   unzip amq-streams-x.y.z-ocp-install-examples.zip
   ```

2.3. INSTALLING AMQ STREAMS

You install AMQ Streams with the Custom Resource Definitions (CRDs) required for deployment.

In this task you create namespaces in the cluster for your deployment. It is good practice to use namespaces to separate functions.

Prerequisites

- Installation requires a user with **cluster-admin** role, such as **system:admin**.

Procedure

1. Login in to the OpenShift cluster using an account that has cluster admin privileges.
   For example:

```
oc login -u system:admin
```

2. Create a new **kafka** (project) namespace for the AMQ Streams Kafka Cluster Operator.

```
oc new-project kafka
```

3. Modify the installation files to reference the new **kafka** namespace where you will install the AMQ Streams Kafka Cluster Operator.

NOTE

By default, the files work in the **myproject** namespace.

- On Linux, use:

```
sed -i 's/namespace: .*/namespace: kafka/' install/cluster-operator/*RoleBinding*.yaml
```

- On Mac, use:

```
sed -i '' 's/namespace: .*/namespace: kafka/' install/cluster-operator/*RoleBinding*.yaml
```

4. Deploy the CRDs and role-based access control (RBAC) resources to manage the CRDs.

```
oc project kafka
oc apply -f install/cluster-operator/
```

5. Create a new **my-kafka-project** namespace where you will deploy your Kafka cluster.

```
oc new-project my-kafka-project
```

6. Give access to **my-kafka-project** to a non-admin user **developer**.
   For example:

```
oc adm policy add-role-to-user admin developer -n my-kafka-project
```

7. Set the value of the *STRIMZI_NAMESPACE* environment variable to give permission to the Cluster Operator to watch the **my-kafka-project** namespace.

```
oc set env deploy/strimzi-cluster-operator STRIMZI_NAMESPACE=kafka,my-kafka-project -n kafka
```

```
oc apply -f install/cluster-operator/020-RoleBinding-strimzi-cluster-operator.yaml -n my-kafka-project
```

```
oc apply -f install/cluster-operator/032-RoleBinding-strimzi-cluster-operator-topic-operator-delegation.yaml -n my-kafka-project
```

```
oc apply -f install/cluster-operator/031-RoleBinding-strimzi-cluster-operator-entity-operator-delegation.yaml -n my-kafka-project
```

The commands create role bindings that grant permission for the Cluster Operator to access the Kafka cluster.

8. Create a new cluster role **strimzi-admin**.

```
oc apply -f install/strimzi-admin
```

9. Add the role to the non-admin user **developer**.

```
oc adm policy add-cluster-role-to-user strimzi-admin developer
```

2.4. CREATING A CLUSTER

With AMQ Streams installed, you create a Kafka cluster, then a topic within the cluster.

When you create a cluster, the Cluster Operator you deployed when installing AMQ Streams watches for new Kafka resources.

Prerequisites

- For the Kafka cluster, ensure a Cluster Operator is deployed.
- For the topic, you must have a running Kafka cluster.

Procedure

1. Log in to the **my-kafka-project** namespace as user **developer**.
   For example:

```
oc login -u developer
oc project my-kafka-project
```

   After new users log in to OpenShift Container Platform, an account is created for that user.

2. Create a new **my-cluster** Kafka cluster with 3 Zookeeper and 3 broker nodes.
   - Use **ephemeral** storage
   - Expose the Kafka cluster outside of the OpenShift cluster using an external listener configured to use **route**.

```
cat << EOF | oc create -f -
apiVersion: kafka.strimzi.io/v1beta1
kind: Kafka
metadata:
  name: my-cluster
spec:
  kafka:
    replicas: 3
    listeners:
      - name: plain
        port: 9092
        type: internal
        tls: false
      - name: tls
        port: 9093
        type: internal
        tls: true
      - name: external
        port: 9094
        type: route (1)
        tls: true
    storage:
      type: ephemeral
  zookeeper:
    replicas: 3
    storage:
      type: ephemeral
  entityOperator:
    topicOperator: {}
EOF
```

3. Wait for the cluster to be deployed:

```
oc wait my-kafka-project/my-cluster --for=condition=Ready --timeout=300s -n kafka
```

4. When your cluster is ready, create a topic to publish and subscribe from your external client. Create the following **my-topic** custom resource definition with 3 replicas and 3 partitions in the **my-cluster** Kafka cluster:

```
cat << EOF | oc create -f -
apiVersion: kafka.strimzi.io/v1beta1
kind: KafkaTopic
metadata:
  name: my-topic
  labels:
    strimzi.io/cluster: "my-cluster"
spec:
  partitions: 3
  replicas: 3
EOF
```

2.5. ACCESSING THE CLUSTER

As **route** is used for external access to the cluster, a cluster CA certificate is required to enable TLS (Transport Layer Security) encryption between the broker and the client.

Prerequisites

- You need a Kafka cluster running within the OpenShift cluster.
- The Cluster Operator must also be running.

Procedure

1. Find the address of the bootstrap **route**:

   ```
   oc get routes my-cluster-kafka-bootstrap -o=jsonpath='{.status.ingress[0].host}{"\n"}'
   ```

   Use the address together with port 443 in your Kafka client as the bootstrap address.

2. Extract the public certificate of the broker certification authority:

   ```
   oc extract secret/my-cluster-cluster-ca-cert --keys=ca.crt --to=- > ca.crt
   ```

3. Import the trusted certificate to a truststore:

   ```
   keytool -keystore client.truststore.jks -alias CARoot -import -file ca.crt
   ```

   You are now ready to start sending and receiving messages.

2.6. SENDING AND RECEIVING MESSAGES FROM A TOPIC

You can test your AMQ Streams installation by sending and receiving messages outside the cluster from **my-topic**.

Use a terminal to run a Kafka producer and consumer on a local machine.

Prerequisites

- Ensure AMQ Streams is installed on the OpenShift cluster.
- ZooKeeper and Kafka must be running to be able to send and receive messages.
- You need a cluster CA certificate for access to the cluster .
- You must be able to access to the latest version of the Red Hat AMQ Streams archive from the AMQ Streams download site .

Procedure

1. Download the latest version of the AMQ Streams archive (**amq-streams-x.y.z-bin.zip**) from the AMQ Streams download site .
   Unzip the file to any destination.

2. Open a terminal, and start the Kafka console producer with the topic **my-topic** and the authentication properties for TLS:

```
bin/kafka-console-producer.sh --broker-list ROUTE-ADDRESS:443 --producer-property
security.protocol=SSL --producer-property ssl.truststore.password=password --producer-
property ssl.truststore.location=./client.truststore.jks --topic my-topic
```

3. Type your message into the console where the producer is running.
4. Press Enter to send the message.
5. Open a new terminal tab or window, and start the Kafka console consumer to receive the messages:

```
bin/kafka-console-consumer.sh --bootstrap-server ROUTE-ADDRESS:443 --consumer-
property security.protocol=SSL --consumer-property ssl.truststore.password=password --
consumer-property ssl.truststore.location=./client.truststore.jks --topic my-topic --from-
beginning
```

6. Confirm that you see the incoming messages in the consumer console.
7. Press Crtl+C to exit the Kafka console producer and consumer.

APPENDIX A. USING YOUR SUBSCRIPTION

AMQ Streams is provided through a software subscription. To manage your subscriptions, access your account at the Red Hat Customer Portal.

Accessing Your Account

1. Go to access.redhat.com.
2. If you do not already have an account, create one.
3. Log in to your account.

Activating a Subscription

1. Go to access.redhat.com.
2. Navigate to **My Subscriptions**.
3. Navigate to **Activate a subscription** and enter your 16-digit activation number.

Downloading Zip and Tar Files

To access zip or tar files, use the customer portal to find the relevant files for download. If you are using RPM packages, this step is not required.

1. Open a browser and log in to the Red Hat Customer Portal **Product Downloads** page at access.redhat.com/downloads.
2. Locate the **Red Hat AMQ Streams** entries in the **JBOSS INTEGRATION AND AUTOMATION** category.
3. Select the desired AMQ Streams product. The **Software Downloads** page opens.
4. Click the **Download** link for your component.

Revised on 2020-12-02 15:51:39 UTC