Red Hat AMQ 7.5

Evaluating AMQ Online on OpenShift

For use with AMQ Online 1.3

Last Updated: 2019-11-06

Red Hat AMQ 7.5 Evaluating AMQ Online on OpenShift

For use with AMQ Online 1.3

Legal Notice

Copyright © 2019 Red Hat, Inc.

The text of and illustrations in this document are licensed by Red Hat under a Creative Commons Attribution–Share Alike 3.0 Unported license ("CC-BY-SA"). An explanation of CC-BY-SA is available at
http://creativecommons.org/licenses/by-sa/3.0/
. In accordance with CC-BY-SA, if you distribute this document or an adaptation of it, you must provide the URL for the original version.

Red Hat, as the licensor of this document, waives the right to enforce, and agrees not to assert, Section 4d of CC-BY-SA to the fullest extent permitted by applicable law.

Red Hat, Red Hat Enterprise Linux, the Shadowman logo, the Red Hat logo, JBoss, OpenShift, Fedora, the Infinity logo, and RHCE are trademarks of Red Hat, Inc., registered in the United States and other countries.

Linux ® is the registered trademark of Linus Torvalds in the United States and other countries.

Java ® is a registered trademark of Oracle and/or its affiliates.

XFS ® is a trademark of Silicon Graphics International Corp. or its subsidiaries in the United States and/or other countries.

MySQL ® is a registered trademark of MySQL AB in the United States, the European Union and other countries.

Node.js ® is an official trademark of Joyent. Red Hat is not formally related to or endorsed by the official Joyent Node.js open source or commercial project.

The OpenStack ® Word Mark and OpenStack logo are either registered trademarks/service marks or trademarks/service marks of the OpenStack Foundation, in the United States and other countries and are used with the OpenStack Foundation's permission. We are not affiliated with, endorsed or sponsored by the OpenStack Foundation, or the OpenStack community.

All other trademarks are the property of their respective owners.

Abstract

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

Table of Contents

CHAPTER 1. INTRODUCTION

1.1. AMQ ONLINE OVERVIEW

Red Hat AMQ Online is an OpenShift-based mechanism for delivering messaging as a managed service. With Red Hat AMQ Online, administrators can configure a cloud-native, multi-tenant messaging service either in the cloud or on premise. Developers can provision messaging using the Red Hat AMQ Console. Multiple development teams can provision the brokers and queues from the Console, without requiring each team to install, configure, deploy, maintain, or patch any software.

AMQ Online can provision different types of messaging depending on your use case. A user can request messaging resources by creating an address space. AMQ Online currently supports two address space types, standard and brokered, each with different semantics. The following diagrams illustrate the high-level architecture of each address space type:

Figure 1.1. Standard address space

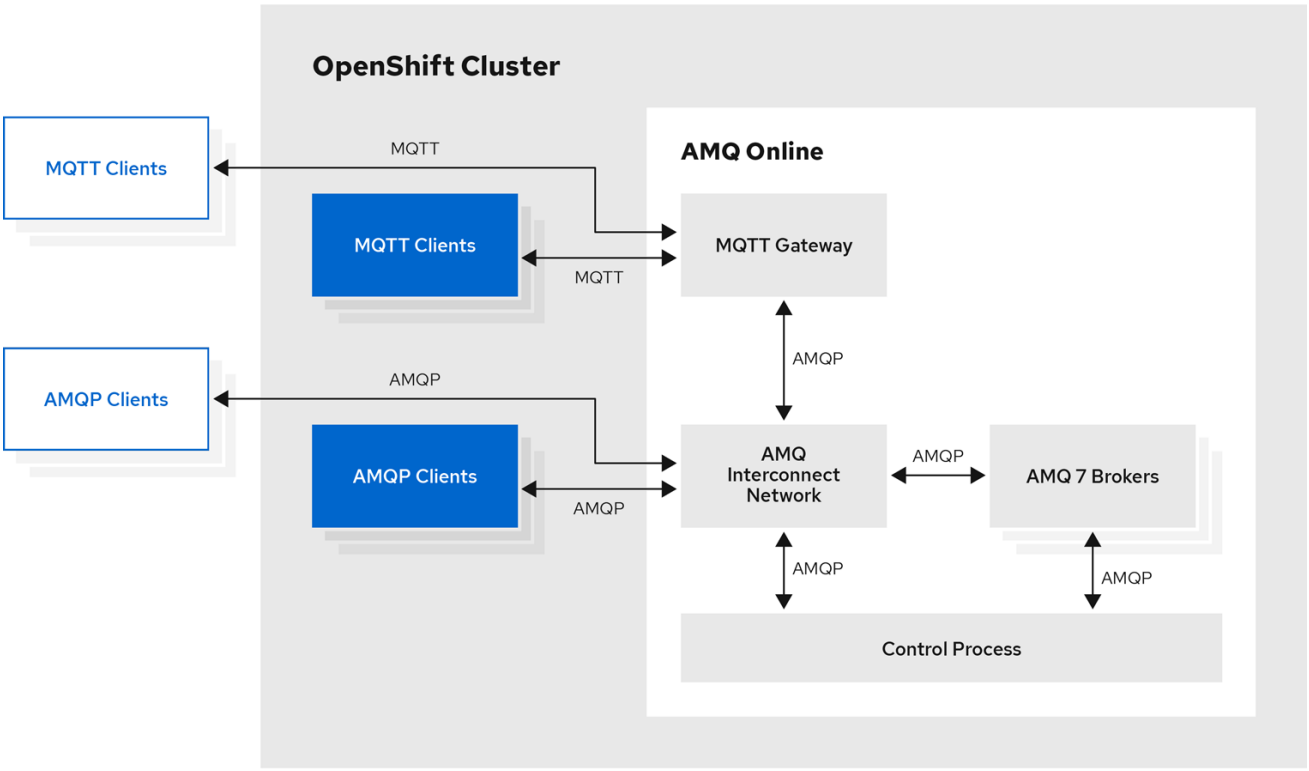
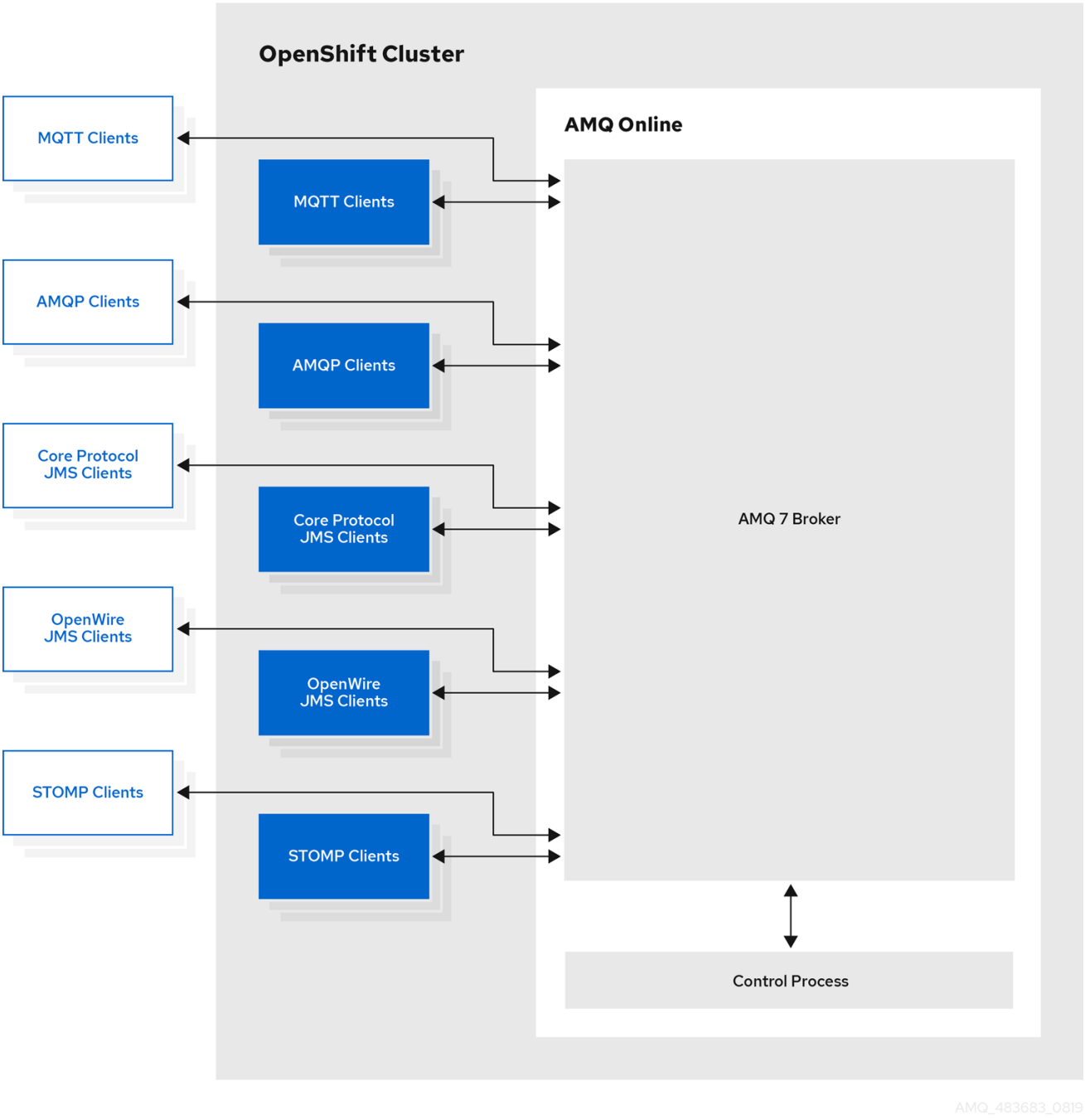


Figure 1.2. Brokered address space



1.2. SUPPORTED FEATURES

The following table shows the supported features for AMQ Online 1.3:

Table 1.1. Supported features reference table

Feature		Brokered address space	Standard address space
Address type	Queue	Yes	Yes
	Topic	Yes	Yes

Feature		Brokered address space	Standard address space
	Multicast	No	Yes
	Anycast	No	Yes
	Subscription	No	Yes
Messaging protocol	AMQP	Yes	Yes
	MQTT	Yes	Technology preview only
	CORE	Yes	No
	OpenWire	Yes	No
	STOMP	Yes	No
Transports	TCP	Yes	Yes
	WebSocket	Yes	Yes
Durable subscriptions	JMS durable subscriptions	Yes	No
	"Named" durable subscriptions	No	Yes
JMS	Transaction support	Yes	No
	Selectors on queues	Yes	No
	Message ordering guarantees (including prioritization)	Yes	No
Scalability	Scalable distributed queues and topics	No	Yes

1.3. SUPPORTED CONFIGURATIONS

For more information about AMQ Online supported configurations see Red Hat AMQ 7 Supported Configurations.

1.4. DOCUMENT CONVENTIONS

1.4.1. Variable text

This document contains code blocks with variables that you must replace with values specific to your installation. In this document, such text is styled as italic monospace.

For example, in the following code block, replace ***my-namespace*** with the namespace used in your installation:

```
sed -i 's/amq-online-infra/my-namespace/' install/bundles/enmasse-with-standard-authservice/*.yaml
```

CHAPTER 2. GETTING STARTED

This guide describes the process of setting up AMQ Online on OpenShift with clients for sending and receiving messages to evaluate its potential use in a production environment.

Prerequisites

- To install AMQ Online, the OpenShift Container Platform command-line interface (CLI) is required.
  - For more information about how to install the CLI on OpenShift 3.x, see the OpenShift Container Platform 3.11 documentation.
  - For more information about how to install the CLI on OpenShift 4.1, see the OpenShift Container Platform 4.1 documentation.
- An OpenShift cluster is required.
- A user on the OpenShift cluster with **cluster-admin** permissions is required to set up the required cluster roles and API services.

2.1. INSTALLING AMQ ONLINE USING A YAML BUNDLE

After completing the download and installation procedures, you must then:

- create an address space
- create an address
- create a messaging user

2.1.1. Downloading AMQ Online

Procedure

- Download and extract the **amq-online-install.zip** file from the AMQ Online download site .

NOTE

Although container images for AMQ Online are available in the Red Hat Container Catalog, we recommend that you use the YAML files provided instead.

2.1.2. Installing AMQ Online using a YAML bundle

The simplest way to install AMQ Online is to use the predefined YAML bundles.

Procedure

1. Log in as a user with **cluster-admin** privileges:

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

2. (Optional) If you want to deploy to a project other than **amq-online-infra** you must run the following command and substitute **amq-online-infra** in subsequent steps:

```
sed -i 's/amq-online-infra/my-project/' install/bundles/amq-online/*.yaml
```

3. Create the project where you want to deploy AMQ Online:

```
oc new-project amq-online-infra
```

4. Change the directory to the location of the downloaded release files.

5. Deploy using the **amq-online** bundle:

```
oc apply -f install/bundles/amq-online
```

6. (Optional) Install the example plans and infrastructure configuration:

```
oc apply -f install/components/example-plans
```

7. (Optional) Install the example roles:

```
oc apply -f install/components/example-roles
```

8. (Optional) Install the **standard** authentication service:

```
oc apply -f install/components/example-authservices/standard-authservice.yaml
```

9. (Optional) Install the Service Catalog integration:

```
oc apply -f install/components/service-broker
oc apply -f install/components/cluster-service-broker
```

2.2. INSTALLING AND CONFIGURING AMQ ONLINE USING THE OPERATOR LIFECYCLE MANAGER

You can use the Operator Lifecycle Manager to install and configure an evaluation instance of AMQ Online.

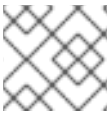
In OpenShift Container Platform 4.1, the Operator Lifecycle Manager (OLM) helps users install, update, and manage the life cycle of all Operators and their associated services running across their clusters. It is part of the Operator Framework, an open source toolkit designed to manage Kubernetes native applications (Operators) in an effective, automated, and scalable way.

The OLM runs by default in OpenShift Container Platform 4.1, which aids cluster administrators in installing, upgrading, and granting access to Operators running on their cluster. The OpenShift Container Platform console provides management screens for cluster administrators to install Operators, as well as grant specific projects access to use the catalog of Operators available on the cluster.

OperatorHub is the graphical interface that OpenShift cluster administrators use to discover, install, and upgrade Operators. With one click, these Operators can be pulled from OperatorHub, installed on the cluster, and managed by the OLM, ready for engineering teams to self-service manage the software in development, test, and production environments.

2.2.1. Installing AMQ Online from the OperatorHub using the OpenShift Container Platform console

You can install the AMQ Online Operator on an OpenShift Container Platform 4.1 cluster by using OperatorHub in the OpenShift Container Platform console.



NOTE

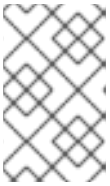
You must install and deploy the AMQ Online Operator in the **openshift-operator** project.

Prerequisites

- Access to an OpenShift Container Platform 4.1 cluster using an account with **cluster-admin** permissions.

Procedure

1. In the OpenShift Container Platform console, log in using an account with **cluster-admin** privileges.
2. Click **Catalog > OperatorHub**.
3. In the **Filter by keyword** box, type **AMQ Online** to find the AMQ Online Operator.
4. Click the AMQ Online Operator. Information about the Operator is displayed.
5. Read the information about the Operator and click **Install**. The Create Operator Subscription page opens.
6. On the **Create Operator Subscription** page, accept all of the default selections and click **Subscribe**.



NOTE

All namespaces on the cluster (default) installs the Operator in the default **openshift-operators** project and makes the Operator available to all projects in the cluster.

The **amq-online** page is displayed, where you can monitor the installation progress of the AMQ Online Operator subscription.

7. After the subscription upgrade status is shown as **Up to date**, click **Catalog > Installed Operators** to verify that the **AMQ Online** ClusterServiceVersion (CSV) is displayed and its **Status** ultimately resolves to **InstallSucceeded** in the **openshift-operators** project.

NOTE

For the **All namespaces...** installation mode, the status resolves to **InstallSucceeded** in the **openshift-operators** project, but the status is **Copied** if you view other projects.

For troubleshooting information, see the OpenShift Container Platform documentation.

Next steps

- Configure AMQ Online using the OpenShift Container Platform console

2.2.2. Configuring AMQ Online using the OpenShift Container Platform console

After installing AMQ Online from the OperatorHub using the OpenShift Container Platform console, create a new instance of a custom resource for the following items within the **openshift-operators** project:

- service infrastructure configuration for an address space type (the example uses the standard address space type)
- an authentication service
- an address space plan
- an address plan

After creating the new instances of the custom resources, next:

- create an address space
- create an address
- create a messaging user

The following procedures use the example data that is provided when using the OpenShift Container Platform console.

2.2.2.1. Creating an infrastructure configuration custom resource using the OpenShift Container Platform console

You must create an infrastructure configuration custom resource to use AMQ Online. This example uses **StandardInfraConfig** for a standard address space.

Procedure

1. From the dropdown menu, select the **openshift-operators** project.
2. Click **Catalog > Installed Operators**.
3. Click **Add > Import YAML** The Import YAML window opens.
4. Copy the following code for **StandardInfraConfig**:

```
apiVersion: admin.enmasse.io/v1beta1
kind: StandardInfraConfig
metadata:
  name: default
```

5. In the Import YAML window, paste the copied code and click **Create**. The StandardInfraConfig Overview page is displayed.
6. Click **Workloads > Pods** In the **Readiness** column, the Pod status is **Ready** when the custom resource has been deployed.

Next steps

- Create an authentication service custom resource using the OpenShift Container Platform console

2.2.2.2. Creating an authentication service custom resource using the OpenShift Container Platform console

You must create a custom resource for an authentication service to use AMQ Online. This example uses the standard authentication service.

Procedure

1. From the dropdown menu, select the **openshift-operators** project.
2. Click **Catalog > Installed Operators**.
3. Click **Add > Import YAML**. The Import YAML window opens.
4. Copy the following code:

```
apiVersion: admin.enmasse.io/v1beta1
kind: AuthenticationService
metadata:
  name: standard-authservice
spec:
  type: standard
```

5. In the Import YAML window, paste the copied code and click **Create**. The AddressPlan Overview page is displayed.
6. Click **Workloads > Pods**. In the **Readiness** column, the Pod status is **Ready** when the custom resource has been deployed.

Next steps

- Create an address space plan custom resource using the OpenShift Container Platform console

2.2.2.3. Creating an address space plan custom resource using the OpenShift Container Platform console

You must create an address space plan custom resource to use AMQ Online. This procedure uses the example data that is provided when using the OpenShift Container Platform console.

Procedure

1. From the dropdown menu, select the **openshift-operators** project.
2. Click **Catalog > Installed Operators**.
3. Click **Add > Import YAML**. The Import YAML window opens.
4. Copy the following code:

```
apiVersion: admin.enmasse.io/v1beta2
```

```
kind: AddressSpacePlan
metadata:
  name: standard-small
spec:
  addressSpaceType: standard
  infraConfigRef: default
  addressPlans:
    - standard-small-queue
  resourceLimits:
    router: 2.0
    broker: 3.0
    aggregate: 4.0
```

5. In the Import YAML window, paste the copied code and click **Create**. The AddressPlan Overview page is displayed.

6. Click **Workloads > Pods**. In the **Readiness** column, the Pod status is **Ready** when the custom resource has been deployed.

Next steps

- Create an address plan custom resource using the OpenShift Container Platform console

2.2.2.4. Creating an address plan custom resource using the OpenShift Container Platform console

You must create an address plan custom resource to use AMQ Online. This procedure uses the example data that is provided when using the OpenShift Container Platform console.

Procedure

1. From the dropdown menu, select the **openshift-operators** project.

2. Click **Catalog > Installed Operators**.

3. Click **Add > Import YAML**. The Import YAML window opens.

4. Copy the following code:

```
apiVersion: admin.enmasse.io/v1beta2
kind: AddressPlan
metadata:
  name: standard-small-queue
spec:
  addressType: queue
  resources:
    router: 0.01
    broker: 0.1
```

5. In the Import YAML window, paste the copied code and click **Create**. The AddressPlan Overview page is displayed.

6. Click **Workloads > Pods**. In the **Readiness** column, the Pod status is **Ready** when the custom resource has been deployed.

Next steps

- Create an address space
- Create an address
- Create a messaging user

2.3. CREATING ADDRESS SPACES USING THE COMMAND LINE

In AMQ Online, you create address spaces using standard command-line tools.

Procedure

1. Log in as a messaging tenant:

```
oc login -u developer
```

2. Create the project for the messaging application:

```
oc new-project myapp
```

3. Create an address space definition:

```
apiVersion: enmasse.io/v1beta1
kind: AddressSpace
metadata:
  name: myspace
spec:
  type: standard
  plan: standard-unlimited
```

4. Create the address space:

```
oc create -f standard-address-space.yaml
```

5. Check the status of the address space:

```
oc get addressspace myspace -o jsonpath={.status.isReady}
```

The address space is ready for use when the previous command outputs **true**.

2.4. CREATING ADDRESSES USING THE COMMAND LINE

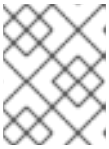
You can create addresses using the command line.

Procedure

1. Create an address definition:

```
apiVersion: enmasse.io/v1beta1
kind: Address
metadata:
```

```
    name: myspace.myqueue
spec:
    address: myqueue
    type: queue
    plan: standard-small-queue
```



NOTE

Prefixing the name with the address space name is required to ensure addresses from different address spaces do not collide.

2. Create the address:

```
oc create -f standard-small-queue.yaml
```

3. List the addresses:

```
oc get addresses -o yaml
```

2.5. CREATING USERS USING THE COMMAND LINE

In AMQ Online users can be created using standard command-line tools.

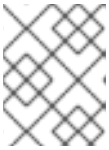
Prerequisites

- You must have already created an address space.

Procedure

1. To correctly base64 encode a password for the user definition file, run the following command:

```
echo -n password | base64 #cGFzc3dvcmQ=
```



NOTE

Be sure to use the **-n** parameter when running this command. Not specifying that parameter will result in an improperly coded password and cause log-in issues.

2. Save the user definition to a file:

```
apiVersion: user.enmasse.io/v1beta1
kind: MessagingUser
metadata:
  name: myspace.user1
spec:
  username: user1
  authentication:
    type: password
    password: cGFzc3dvcmQ= # Base64 encoded
  authorization:
    - addresses: ["myqueue", "queue1", "queue2", "topic*"]
```

```
    operations: ["send", "recv"]
  - addresses: ["anycast1"]
    operations: ["send"]
```

3. Create the user and associated user permissions:

```
oc create -f user-example1.yaml
```

4. Confirm that the user was created:

```
oc get messagingusers
```

2.6. SENDING AND RECEIVING MESSAGES

Prerequisites

- Installed Apache Qpid Proton Python bindings.
- An address space named **myspace** must be created.
- An address named **myqueue** must be created.
- A user named **user1** with password **password** must be created.

Procedure

1. Save Python client example to a file:

```
from __future__ import print_function, unicode_literals
import optparse
from proton import Message
from proton.handlers import MessagingHandler
from proton.reactor import Container

class HelloWorld(MessagingHandler):
    def __init__(self, url):
        super(HelloWorld, self).__init__()
        self.url = url

    def on_start(self, event):
        event.container.create_receiver(self.url)
        event.container.create_sender(self.url)

    def on_sendable(self, event):
        event.sender.send(Message(body="Hello World!"))
        event.sender.close()

    def on_message(self, event):
        print("Received: " + event.message.body)
        event.connection.close()

parser = optparse.OptionParser(usage="usage: %prog [options]")
parser.add_option("-u", "--url", default="amqps://localhost:5672/myqueue",
                  help="url to use for sending and receiving messages")
```

```
opts, args = parser.parse_args()

try:
    Container(HelloWorld(opts.url)).run()
except KeyboardInterrupt: pass
```

2. Retrieve the address space messaging endpoint host name:

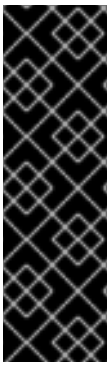
```
oc get addressspace myspace -o 'jsonpath={.status.endpointStatuses[?
(@.name=="messaging")].externalHost}'
```

Use the output as the host name in the following step.

3. Run the client:

```
python client-example1.py -u
amqps://user1:password@messaging.example1.com:443/myqueue
```

CHAPTER 3. GETTING STARTED WITH INTERNET OF THINGS (IOT) ON AMQ ONLINE

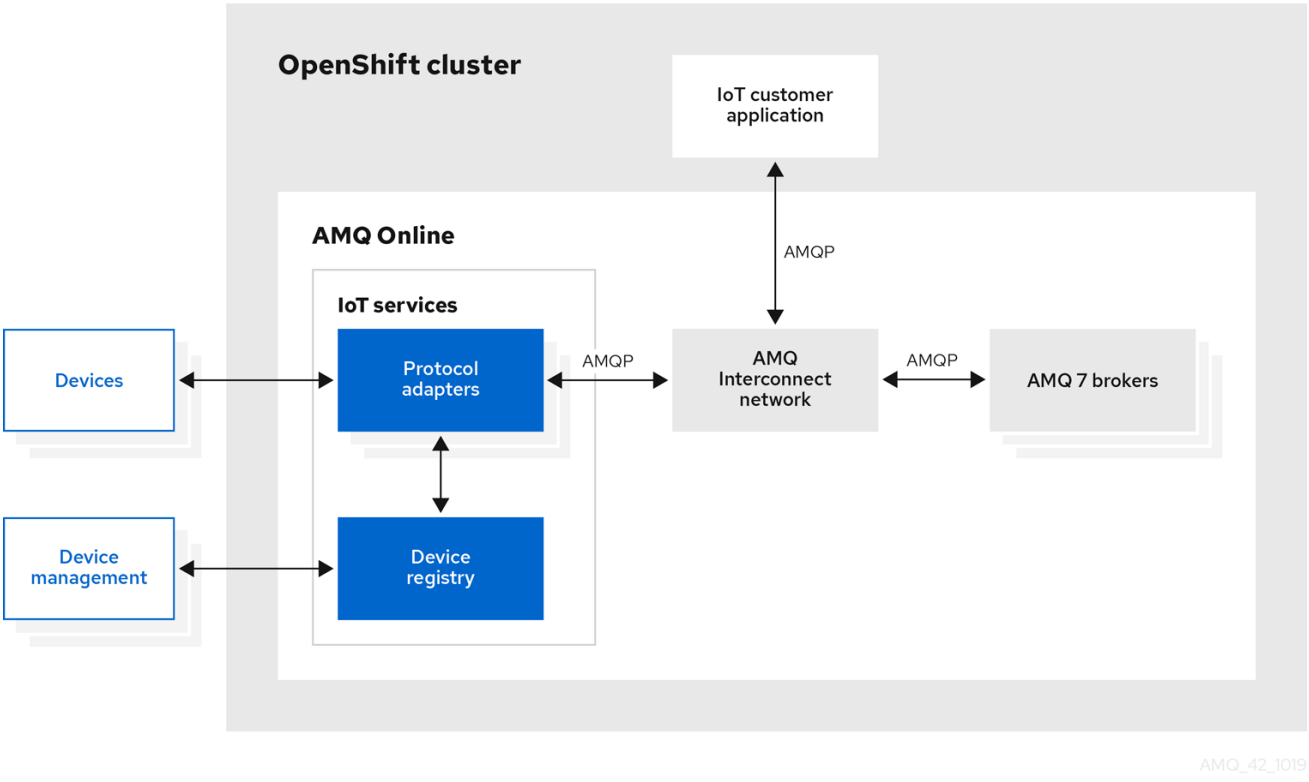


IMPORTANT

The Internet of Things (IoT) feature of AMQ Online is a Technology Preview feature only. Technology Preview features are not supported with Red Hat production service-level agreements (SLAs) and might not be functionally complete; therefore, Red Hat does not recommend implementing any Technology Preview features in production environments. This Technology Preview feature provides early access to upcoming product innovations, enabling you to test functionality and provide feedback during the development process. For more information about support scope, see Technology Preview Features Support Scope.

3.1. IOT CONNECTIVITY CONCEPTS

The Internet of Things (IoT) connectivity feature enables AMQ Online to be used for managing and connecting devices with back-end applications. In a typical IoT application, devices have different requirements than ordinary messaging applications. Instead of using arbitrary addresses and security configurations that are typically available, developers can use the IoT services to handle device identities and security configurations explicitly, support multiple protocols often used in the IoT space, and provide uniform support for expected device communication patterns.



One of the key concepts is a *device registry*, which developers use to register devices and provide their credentials. With these credentials, devices can then connect to *protocol adapters* using one of the supported protocols: HTTP, MQTT, LoRaWAN, and SigFox. Once connected, devices can send and receive messages from back-end applications using one of the following messaging semantics:

- Telemetry: Allows devices to send non-durable data to back-end applications, so messages are sent using the **multicast** address type. This option is best for sending non-critical sensor readings.

- Events: Allows devices to send durable data to the back-end applications, so messages are sent using the **queue** address type. This option is best for sending more important device data such as alerts and notifications.

Back-end applications can also send **Command** messages to devices. Commands can be used to trigger actions on devices. Examples include updating a configuration property, installing a software component, or switching the state of an actuator.

3.2. INSTALLING AMQ ONLINE USING A YAML BUNDLE

The simplest way to install AMQ Online is to use the predefined YAML bundles.

Procedure

1. Log in as a user with **cluster-admin** privileges:

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

2. (Optional) If you want to deploy to a project other than **amq-online-infra** you must run the following command and substitute **amq-online-infra** in subsequent steps:

```
sed -i 's/amq-online-infra/my-project/' install/bundles/amq-online/*.yaml
```

3. Create the project where you want to deploy AMQ Online:

```
oc new-project amq-online-infra
```

4. Change the directory to the location of the downloaded release files.

5. Deploy using the **amq-online** bundle:

```
oc apply -f install/bundles/amq-online
```

6. (Optional) Install the example plans and infrastructure configuration:

```
oc apply -f install/components/example-plans
```

7. (Optional) Install the example roles:

```
oc apply -f install/components/example-roles
```

8. (Optional) Install the **standard** authentication service:

```
oc apply -f install/components/example-authservices/standard-authservice.yaml
```

9. (Optional) Install the Service Catalog integration:

```
oc apply -f install/components/service-broker
oc apply -f install/components/cluster-service-broker
```

3.3. INSTALLING IOT SERVICES

To get started using the IoT feature on AMQ Online, you must first install the IoT services.

Procedure

1. (Optional) If you want to deploy to a project other than **amq-online-infra** you must run the following command and substitute **amq-online-infra** in subsequent steps:

```
sed -i 's/amq-online-infra/my-project/' install/preview-bundles/iot/*.yaml
```

2. Deploy the IoT bundles:

```
oc apply -f install/preview-bundles/iot
```

3. Create certificates for the MQTT protocol adapter. For testing purposes, you can create a self-signed certificate:

```
./install/components/iot/examples/k8s-tls/create
oc create secret tls iot-mqtt-adapter-tls --key=install/components/iot/examples/k8s-tls/build/iot-mqtt-adapter-key.pem --cert=install/components/iot/examples/k8s-tls/build/iot-mqtt-adapter-fullchain.pem
```

NOTE

If your cluster is not running on **localhost**, you need to specify the cluster host name when creating certificates to allow external clients (like MQTT) to properly connect to the appropriate services. For example:

```
CLUSTER=x.x.x.x.nip.io install/components/iot/examples/k8s-tls/create
```

4. Install the Red Hat Data Grid server:

```
oc apply -f install/components/iot/examples/infinispan/common
oc apply -f install/components/iot/examples/infinispan/manual
```

5. Install an example IoT infrastructure configuration:

```
oc apply -f install/components/iot/examples/iot-config.yaml
```

3.4. CREATING AN IOT PROJECT

After installing the IoT services, you create an IoT project.

Prerequisites

- The IoT services are installed .

Procedure

1. Log in as a messaging tenant:

```
oc login -u developer
```

2. Create a *managed* IoT project:

```
oc new-project myapp
oc create -f install/components/iot/examples/iot-project-managed.yaml
```

3. Wait for the resources to be ready:

```
oc get addressspace iot
oc get iotproject iot
```

NOTE

Make sure that the **Phase** field shows **Ready** status for both resources.

4. Create a messaging consumer user:

```
oc create -f install/components/iot/examples/iot-user.yaml
```

3.5. CREATING AN IOT DEVICE

After installing the IoT services and creating an IoT project, you can create an IoT device for the device you want to monitor.

Prerequisites

- The IoT services are installed .
- An IoT project is created.

3.5.1. Registering a new device

Procedure

1. Export the device registry host:

```
export REGISTRY_HOST=$(oc -n enmasse-infra get routes device-registry --template='{{
.spec.host }}')
```

2. Export the device registry access token:

```
export TOKEN=$(oc whoami --show-token)
```

This token is used to authenticate against the device registry management API.

3. Register a device with a defined ID (this example uses **4711**):

```
curl --insecure -X POST -i -H 'Content-Type: application/json' -H "Authorization: Bearer
${TOKEN}" https://$REGISTRY_HOST/v1/devices/myapp.iot/4711
```

4. (Optional) If you need to provide additional registration information, do so as follows:

```
curl --insecure -X POST -i -H 'Content-Type: application/json' -H "Authorization: Bearer
${TOKEN}" --data-binary '{
 "via": ["gateway1"]
}' https://$REGISTRY_HOST/v1/devices/myapp.iot/4711
```

3.5.2. Setting user name and password credentials for a device

Procedure

1. Add the credentials for a device:

```
curl --insecure -X PUT -i -H 'Content-Type: application/json' -H "Authorization: Bearer
${TOKEN}" --data-binary '[{
 "type": "hashed-password",
 "auth-id": "sensor1",
 "secrets": [{
  "pwd-plain":"'hono-secret'"
 }]
}]' https://$REGISTRY_HOST/v1/credentials/myapp.iot/4711
```

3.6. SENDING AND RECEIVING TELEMETRY DATA

3.6.1. Starting the telemetry consumer

Prerequisites

- The IoT services are installed .
- An IoT project is created.
- An IoT device is created.

Procedure

1. Download the Eclipse Hono command-line client .
2. Get the messaging endpoint certificate:

```
oc -n myapp get addressspace iot -o jsonpath={.status.caCert} | base64 --decode > tls.crt
```

3. Export the messaging endpoint host and port:

```
export MESSAGING_HOST=$(oc -n myapp get addressspace iot -o jsonpath=
{.status.endpointStatuses[?\(@.name==\'messaging\'\)].externalHost})
export MESSAGING_PORT=443
```

4. Run the consumer application:

```
java -jar hono-cli-*-exec.jar --hono.client.host=$MESSAGING_HOST --
hono.client.port=$MESSAGING_PORT --hono.client.username=consumer --
hono.client.password=foobar --tenant.id=myapp.iot --hono.client.trustStorePath=tls.crt --
message.type=telemetry
```

3.6.2. Sending telemetry data

Procedure

1. Send a message using HTTP protocol:

```
curl --insecure -X POST -i -u sensor1@myapp.iot:hono-secret -H 'Content-Type:
application/json' --data-binary '{"temp": 5}' https://$(oc -n enmasse-infra get routes iot-http-
adapter --template='{{ .spec.host }}')/telemetry
```

2. Send a message using MQTT protocol:

```
mosquitto_pub -d -h $(oc -n enmasse-infra get routes iot-mqtt-adapter --template='{{
.spec.host }}') -p 443 -u 'sensor1@myapp.iot' -P hono-secret -t telemetry -m '{"temp": 5}' -i
4711 --cafile install/components/iot/examples/k8s-tls/build/iot-mqtt-adapter-fullchain.pem
```

3.7. SENDING AND RECEIVING EVENT DATA

3.7.1. Starting the event consumer

Prerequisites

- The IoT services are installed .
- An IoT project is created.
- An IoT device is created.

Procedure

1. Download the Eclipse Hono command-line client .
2. Get the messaging endpoint certificate:

```
oc -n myapp get addressspace iot -o jsonpath={.status.caCert} | base64 --decode > tls.crt
```

3. Export the messaging endpoint host and port:

```
export MESSAGING_HOST=$(oc -n myapp get addressspace iot -o jsonpath=
{.status.endpointStatuses[?\(@.name==\'messaging\'\)].externalHost})
export MESSAGING_PORT=443
```

4. Run the consumer application:

```
java -jar hono-cli-*-exec.jar --hono.client.host=$MESSAGING_HOST --
hono.client.port=$MESSAGING_PORT --hono.client.username=consumer --
hono.client.password=foobar --tenant.id=myapp.iot --hono.client.trustStorePath=tls.crt --
message.type=event
```

3.7.2. Sending event data

Procedure

1. Send a message using HTTP protocol:

```
curl --insecure -X POST -i -u sensor1@myapp.iot:hono-secret -H 'Content-Type:
application/json' --data-binary '{"temp": 5}' https://$(oc -n enmasse-infra get routes iot-http-
adapter --template='{{ .spec.host }}')/event
```

2. Send a message using MQTT protocol:

```
mosquitto_pub -d -h $(oc -n enmasse-infra get routes iot-mqtt-adapter --template='{{
.spec.host }}') -p 443 -u 'sensor1@myapp.iot' -P hono-secret -t event -m '{"temp": 5}' -i 4711 -
-cafile install/components/iot/examples/k8s-tls/build/iot-mqtt-adapter-fullchain.pem
```

CHAPTER 4. UNINSTALLING AMQ ONLINE

You must uninstall AMQ Online using the same method that you used to install AMQ Online.

4.1. UNINSTALLING AMQ ONLINE USING THE YAML BUNDLE

This method uninstalls AMQ Online that was installed using the YAML bundle.

Procedure

1. Log in as a user with **cluster-admin** privileges:

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

2. Delete the cluster-level resources:

```
oc delete clusterrolebindings -l app=enmasse
oc delete crd -l app=enmasse
oc delete clusterroles -l app=enmasse
oc delete apiservices -l app=enmasse
oc delete oauthclients -l app=enmasse
```

3. (Optional) Delete the service catalog integration:

```
oc delete clusterservicebrokers -l app=enmasse
```

4. Delete the project where AMQ Online is deployed:

```
oc delete project amq-online-infra
```

4.2. UNINSTALLING THE AMQ ONLINE OPERATOR USING THE OPENSHIFT CONTAINER PLATFORM 4.X CONSOLE

You can uninstall the AMQ Online Operator on an OpenShift Container Platform 4.1 cluster in the OpenShift Container Platform console.

Prerequisites

- An installed AMQ Online Operator on a OpenShift Container Platform 4.1 cluster.

Procedure

1. From the Project list, select the **openshift-operators** project.
2. Click **Catalog → Operator Management**. The Operator Management page opens.
3. Click the **Operator Subscriptions** tab.
4. Find the AMQ Online Operator you want to uninstall. In the far right column, click the vertical ellipsis icon and select **Remove Subscription**.

5. When prompted by the Remove Subscription window, select the **Also completely remove the AMQ Online Operator from the selected namespace** check box to remove all components related to the installation.

6. Click **Remove**. The AMQ Online Operator will stop running and no longer receive updates.

Next steps

- To completely remove all remaining resources, see Removing remaining resources after uninstalling AMQ Online using the Operator Lifecycle Manager.

4.2.1. Removing remaining resources after uninstalling AMQ Online using the Operator Lifecycle Manager

Due to ENTMQMAAS-1281, some resources remain after uninstalling AMQ Online using the Operator Lifecycle Manager. This procedure removes the remaining resources, which completely uninstalls AMQ Online.

Prerequisites

- Uninstalled the AMQ Online Operator using the OpenShift Container Platform 4.x console

Procedure

1. On the command line, log in as a user with permissions to run commands in the **openshift-operators** project:

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

2. Change to the **openshift-operators** project:

```
oc project openshift-operators
```

3. Run the following commands to remove any remaining resources:

```
oc delete all -l app=enmasse
oc delete crd -l app=enmasse
oc delete apiservices -l app=enmasse
oc delete cm -l app=enmasse
oc delete secret -l app=enmasse
```

APPENDIX A. USING YOUR SUBSCRIPTION

AMQ Online 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 Red Hat 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 Online** entries in the **JBOSS INTEGRATION AND AUTOMATION** category.
3. Select the desired AMQ Online product. The Software Downloads page opens.
4. Click the **Download** link for your component.

Registering your system for packages

To install RPM packages on Red Hat Enterprise Linux, your system must be registered. If you are using zip or tar files, this step is not required.

1. Go to access.redhat.com.
2. Navigate to **Registration Assistant**.
3. Select your OS version and continue to the next page.
4. Use the listed command in your system terminal to complete the registration.

To learn more see How to Register and Subscribe a System to the Red Hat Customer Portal .

Revised on 2019-11-06 18:46:07 UTC