



Red Hat JBoss A-MQ 6.2

Cloud Deployment Guide

Centrally configure and provision assets in the cloud

Red Hat JBoss A-MQ 6.2 Cloud Deployment Guide

Centrally configure and provision assets in the cloud

JBoss A-MQ Docs Team

Content Services

fuse-docs-support@redhat.com

Legal Notice

Copyright © 2015 Red Hat.

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, 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 Software Collections 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 use the JBoss A-MQ Fabric cloud APIs to provision, configure, and deploy applications into cloud environments.

Table of Contents

CHAPTER 1. OPENSIFT ENTERPRISE	3
1.1. OVERVIEW	3
1.2. MINIMUM GEAR REQUIREMENTS	3
1.3. GETTING STARTED	5
1.4. PORT CONFIGURATION	8
1.5. FUSE BUILDER CARTRIDGE	11
1.6. FABRIC MANAGEMENT	12
1.7. HIGH AVAILABILITY	14
1.8. UPGRADING THE JBOSS A-MQ CARTRIDGE	15

CHAPTER 1. OPENSIFT ENTERPRISE

Abstract

This section explains how to get started in the cloud with OpenShift Enterprise and Red Hat JBoss A-MQ.

1.1. OVERVIEW

Basic technologies

This OpenShift Enterprise tutorial is based on the following technology stack:

- *OpenShift Platform as a Service (PaaS)*. Red Hat's OpenShift PaaS provides developers with the capability to develop, host, and scale applications in a cloud environment. You can choose a public, private, or hybrid cloud environment. In this tutorial we use OpenShift Enterprise to deploy JBoss A-MQ.
- *Red Hat JBoss A-MQ 6.2*. In this tutorial we use version 6.2 of JBoss A-MQ.

OpenShift applications

[OpenShift](#) is an open source PaaS that enables you to develop, deploy and host applications in a cloud environment. Before you build an application, you create a cartridge that hosts your application code and dependencies. You can choose a QuickStart cartridge, upload your own source code, or link to source code from a public repository.

The default JBoss A-MQ cartridge hosts the JBoss A-MQ application and runs the Fuse Management Console container. Each additional container that you create in JBoss A-MQ appears as an application on the OpenShift **Applications** page.

Fuse Management Console

After you create the JBoss A-MQ cartridge, you receive a URL based on the OpenShift application information that you provided. Use this URL to open the Fuse Management Console. The management console shows information about all containers, fabrics, and dependencies for the JBoss A-MQ cartridge.

1.2. MINIMUM GEAR REQUIREMENTS

This section describes the resource requirements for the JBoss A-MQ cartridge. When you install new OpenShift nodes, you can choose the preconfigured xPaaS gear profile. If you configure existing nodes, you use the xPaaS gear profile from the file that OpenShift provides.

To view the full property descriptions and additional information about the xPaaS gear profile, see the following file: `/etc/openshift/resource_limits.conf.xpaas.m3.xlarge`

For general information about how to configure gears on OpenShift Enterprise, see the section [Gear Profiles](#) in the OpenShift Enterprise documentation.

Basic gear properties

The following table lists a summary of the basic gear properties and values to set when you want to use an xPaaS cartridge:

Property	Value
node_profile	xpaas
quota_blocks	5242880
max_active_gears	50
no_overcommit_active	false
limits_noproc	2142
cpu_shares	128
cpu_cfs_quota_us	200000
memory_limits_in_bytes	1073741824
memory_memsw_limit_in_bytes	1610612736
memory_move_charge_at_immigrate	1
memory_oom_control	1
max_active_gears	50

Additional gear properties

The following table lists additional properties and alternate values for some of the gear properties to use when the gear is throttled, frozen, thawed, or boosted:

Template	Property	Value
Throttle	cpu_shares	128
	cpu_cfs_quota_us	100000
	apply_period	120
	apply_percent	30
	restore_percent	70
Freeze	freezer_state	FROZEN

Template	Property	Value
Thaw	freezer_state	THAWED
Boost	cpu_shares	256
	cpu_cfs_quota_us	400000

1.3. GETTING STARTED

This section describes how to configure your OpenShift Enterprise environment and install JBoss A-MQ.

1.3.1. Install the xPaaS Gear Profile on OpenShift Enterprise Nodes

This section describes how to configure OpenShift Enterprise nodes to support JBoss A-MQ gear profiles after you install each node.

Alternatively, you can specify the xPaaS profile when you install each new node to configure the new node with the required gear properties.



NOTE

You must install the xPaaS gear profile on each node where you intend to deploy the JBoss A-MQ cartridge. The nodes cannot be associated with districts or contain other gears.

1. Edit the `/etc/openshift/resource_limits.conf` file.
 - Replace the contents of the file with the contents of the `/etc/openshift/resource_limits.conf.xpaas.m3.xlarge` file.
 - To prevent memory overcommit, set the maximum number of gears in the `max_active_gears` property to the number of gears that the node can support. The number of active gears needs to reflect the available RAM and disk space of the node.
2. Edit the `/etc/openshift/node.conf` file.
 - Add the following property: **PORTS_PER_USER=15**
 - Add the following value to the **OPENSIFT_FRONTEND_HTTP_PLUGINS** property:


```
openshift-origin-frontend-haproxy-sni-proxy
```
3. In the `/etc/openshift/node-plugins.d/openshift-origin-frontend-haproxy-sni-proxy.conf` file, update the list of the **PROXY_PORTS** property to include ten ports as follows:


```
PROXY_PORTS="2303, 2304, 2305, 2306, 2307, 2308, 2309, 2310, 2311, 2312"
```
4. Configure your firewall to allow TCP/SSL traffic through the ports that you specified in the **PROXY_PORTS** property.

For example, if you use the `/etc/sysconfig/iptables` to manage your firewall configuration, add the following line before the last instance of the **-A INPUT** rule:

```
-A INPUT -m state --state NEW -m tcp -p tcp --dport <port_number>:  
<port_number> -j ACCEPT
```

The following example shows the port configuration for the ports numbered 2303 to 2313:

```
-A INPUT -m state --state NEW -m tcp -p tcp --dport 2303:2312 -j  
ACCEPT
```

After you edit this file, run the **service iptables reload** command to apply the change.

5. Restart the **openshift-sni-proxy** service.

1.3.2. Install the JBoss A-MQ Cartridge on OpenShift Enterprise

This section describes how to install the JBoss A-MQ cartridge on OpenShift Enterprise. You must install the cartridge on each node in your OpenShift Enterprise domain, regardless of whether you intend to deploy JBoss A-MQ applications on that node.

The JBoss A-MQ cartridges is shipped as an RPM package in an erratum that you apply in the same way that you apply asynchronous errata updates to OpenShift Enterprise.

For general information about OpenShift errata updates, see the [Asynchronous Errata Updates](#) section of the OpenShift Enterprise documentation.

1. Download the advisory that contains the JBoss A-MQ cartridge. See the [OpenShift Enterprise 2 General Advisories](#) page on the Red Hat Customer Portal for a list of all OpenShift Enterprise 2 errata.
2. On each node, install the cartridge RPM package with **yum** in the same way you install other OpenShift Enterprise components.
3. Restart each node with the following command:

```
service ruby193-mcollective restart
```

1.3.3. Configure the OpenShift Enterprise Broker to Support xPaaS Gears

This section describes how to add the xPaaS gear profiles to the OpenShift Enterprise broker and how to create a district for the gear size.

For general information on districts in the OpenShift Enterprise broker, see the [Managing Districts](#) section in the OpenShift Enterprise Deployment Guide and [Capacity Planning and Districts](#) section in the OpenShift Enterprise Administration Guide.

1. On the OpenShift Enterprise broker, import the installed cartridges with the following command:

```
oo-admin-ctl-cartridge -c import-profile --activate
```

2. Edit the `/etc/openshift/broker.conf` file.
 - Add the value **xpaas** to the `VALID_GEAR_SIZES` property.

- Add the value **xpaas** to the `DEFAULT_GEAR_CAPABILITIES` property.

You must restart the **openshift-broker** service after you edit this file.

3. Run the following commands:

```
oo-admin-ctl-user -c -l <user_name> --addgearsizes xpaas
```

Grants existing OpenShift Enterprise users permissions to create xPaaS gears.

```
oo-admin-ctl-district -c create -n <district_name> -p xpaas
```

Creates a district for the xPaaS nodes.

```
oo-admin-ctl-district -c remove-capacity -n <district_name> -s 4000
```

Sets the district capacity to a maximum of 2,000 gears.

```
oo-admin-ctl-district -c add-node -n <district_name> -i <node_hostname>
```

Adds the xPaaS nodes to the district.

1.3.4. Create the JBoss A-MQ Application

You create the JBoss A-MQ application from the OpenShift Enterprise management console or from the command line, in the same way that you create other applications on the node.

For general information on how to create applications in OpenShift Enterprise, see the [Creating an Application](#) section in the OpenShift Enterprise documentation.

After successful creation of A-MQ Application, the terminal displays information for the newly created application. You can use the provided URL to access A-MQ web-based admin console. Enter the login credentials that are provided to you after the application creation.

Choosing the xPaaS gear profile

When you create the JBoss A-MQ application, make sure to specify the xPaaS gear profile:

- If you create the application from the command line, include the **-g xpaas** option in the command. For example:

```
rhc create-app <app_name> amq-6.2.0 -g xpaas
```

- If you create the application from the OpenShift Enterprise management console, choose **xPaaS** from the **Gear Size** drop-down list.

1.3.5. Deploy Quickstarts

This section describes how to deploy a JBoss A-MQ quickstart to run in the OpenShift Enterprise domain. Normally, you run the **mvn clean install** command to build the quickstart. However, in the OpenShift Enterprise domain, you can deploy the quickstart by assigning the quickstart profile to the container. For example, the profile for cbr quickstart is located at:

```
/hawtio/index.html#/wiki/branch/1.0/view/fabric/profiles/example/quickstarts/  
cbr.profile
```

After you assign the cbr profile, the **Camel** tab will be activated. You can view the quickstart attributes by clicking Camel tab. The **Logs** tab will show the messages related to cbr quickstart.

For more information regarding Profiles, see [Management Console User Guide](#).

1.4. PORT CONFIGURATION

This section describes how to configure, map, and assign ports when you want to connect to the JBoss A-MQ application.

1.4.1. Choosing SSL or Non-SSL Ports

When you deploy your JBoss A-MQ application you can choose to use an SSL connection or non-SSL connection.

SSL connection

This connection uses static predefined ports to connect to the JBoss A-MQ application. SSL connections are slower than non-SSL connections due to processing overhead at run-time, but you can determine the port number to use at run-time when you first install the application.

Non-SSL connection

This connection uses a dynamic port number that OpenShift Enterprise allocates based on the available ports when you install the JBoss A-MQ application. After you install the application, you must determine which port numbers the clients need to use to connect to the application.

Configuring an SSL connection

1. In the ActiveMQ JMS client, specify the SSL port number in the **ActiveMQConnectionFactory** property. By default, the following port numbers are available for SSL connections:

Openwire

2303

STOMP

2304

AMQP 1.0

2305

MQTT 3.1

2306

2. Copy the contents of the self-signed public server certificate to a file named **server.crt** and store the file in your local machine. You can access the certificate with the URL that appears when you first install the JBoss A-MQ application, or from the default profile directory in the Wiki tab of the Fuse Management Console.
3. Run the following command to create a Java keystore that imports the certificate:

```
$ keytool -importcert -keystore my.jks -storepass password \
-file server.crt -noprompt
```

4. Configure the JVM to use the keystore when the client connects to the application:

```
$ java -Djavax.net.ssl.trustStore=my.jks ...
```

Configuring a non-SSL connection

1. After you install the JBoss A-MQ cartridge, run one of the following commands:

```
$echo ${OPENSIFT_AMQ_OPENWIRE_PROXY_PORT}
$echo ${OPENSIFT_AMQ_MQTT_PROXY_PORT}
$echo ${OPENSIFT_AMQ_AMQP_PROXY_PORT}
$echo ${OPENSIFT_AMQ_STOMP_PROXY_PORT}
```

2. Specify the port number that the broker returns in the connection URL. For example:

```
tcp://amq-demo.openshift.example.com:63373
```

1.4.2. Port Binding

Some Camel components and CXF endpoints must bind to specific ports to enable client connections. When you configure the JBoss A-MQ cartridge you must bind components such as camel-netty to these ports.

You can use the following system properties variables to bind components to private ports:

- **app1.port**
- **app2.port**
- **app3.port**



NOTE

If you deploy an ActiveMQ container, the **app1.port** system property is reserved for the container.

You specify the port system property in the connection properties with the following format:

```
${bind.address}:${system_property}
```

To bind a component to a public port, you use the following connection address format:

```
${publichostname}:${app1.public.port}
```

1.4.3. Public Port Mapping

The JBoss A-MQ cartridge includes the PublicPortMapper tool that translates private ports in CXF endpoint addresses to public ports. This tool ensures that users can connect to the JBoss A-MQ application from outside the OpenShift Enterprise domain without exposing the private ports that CXF

requires to run.

The following CXF components use the PublicPortMapper tool:

io.fabric8.cxf.registry.FabricCxfRegistrationHandler

This handler uses the PublicPortMapper tool to translate CXF endpoint addresses. The tool maps the port for each endpoint based on the **address** property of the **jaxws:server** element:

```
<jaxws:server id="service1" serviceClass="io.fabric8.demo.cxf.Hello"
              address="http://$[bind.address]:$[app1.port]/server/server1">
```

The following example shows the source address of a CXF endpoint:

```
http://127.2.123.129:3001/server/server1
```

The following example shows the translated external URL:

```
http://app-domain.openshift.com:47106/server/server1
```

The tool writes the translated address to one of the following ZooKeeper paths:

- **/fabric/registry/clusters/apis/rest/{name}/{version}/{container}**
- **/fabric/registry/clusters/apis/ws/{name}/{version}/{container}**

io.fabric8.cxf.FabricLoadBalancerFeature

This feature uses the PublicPortMapper tool to translate the addresses of all endpoints in the cluster. The tool maps the ports based on the list of addresses in the **group** array property of the feature.

Each time the **jaxws:server** component starts, the **io.fabric8.cxf.FabricServerListener** service retrieves the addresses from all active endpoints and stores the addresses in the **group** property. The feature then invokes the PublicPortMapper tool to translate the addresses to external connection URLs.

The tool writes the addresses to the following ZooKeeper path:

```
/fabric/cxf/endpoints/<path-configured-for-FabricLoadBalancerFeature>
```

io.fabric8.camel.FabricPublisherEndpoint

This endpoint uses the PublicPortMapper tool to translate the address of the listener based on the **from uri** property of the **io.fabric8.camel.FabricComponent** route.

The following example shows the source address of a Jetty listener:

```
<from uri="fabric-camel:cluster:jetty:http://0.0.0.0:[[port]]/fabric"/>
```

The tool writes the translated address to the following ZooKeeper path:

```
/fabric/clusters/fabric/registry/camel/endpoints/cluster/<cluster_instance_number>
```

The following example shows the translated address:

```
http://fuse0-test.openshift.example.com:40257/fabric
```

1.5. FUSE BUILDER CARTRIDGE

Overview

The Fuse Builder cartridge builds the Maven repository of a JBoss A-MQ application, and rebuilds the repository each time you update any of the repository artifacts.

This cartridge provides an HTTP connection to the repository that you can use to connect to the repository from all nodes that run JBoss A-MQ applications.

When you deploy JBoss A-MQ applications in a high availability configuration, you can specify this cartridge as the remote Maven repository to ensure that the master and slave nodes can always access the Maven artifacts.

Installing the cartridge

The Fuse Builder cartridge is shipped as an RPM package. You install the cartridge in the same way that you install the JBoss A-MQ cartridge.

When you install this cartridge, note the following guidelines:

- You must deploy at least one JBoss A-MQ cartridge in the OpenShift Enterprise domain before you install and deploy this cartridge.
- You must install this cartridge on every node in the OpenShift Enterprise domain.
- You can install this cartridge with any gear profile.

Configuring security

Before you begin to use the Fuse Builder cartridge, you must specify which users can download the artifacts from the Maven repository.

1. Clone the cartridge Git repository to your development machine.
2. In the **.openshift/config** directory of the cloned repository, open the **httpd.conf** file and uncomment the security section.
3. Run the following command to create a password file in the **.openshift/config** directory:

```
htpasswd -cb passwords <USERNAME> <PASSWORD>
```

4. Commit and push the new password file and the edited **httpd.conf** file to the remote repository.

Adding the Maven repository to the JBoss A-MQ application

In each JBoss A-MQ application that you want to connect with the Maven repository, add the repository address to the default profile.

If you use the Fuse Management Console, you access the default profile with the following path:

```
/hawtio/index.html#/wiki/branch/<version_number>/view/fabric/profiles/default.profile/io.fabric8.agent.properties
```

You add the repository URL to the list of Maven repositories in the **org.ops4j.pax.url.mvn.repositories** property.

The URL pattern must be in one of the following formats:

- **http://\${app-dns}/repo**
- **https://\${user}:\${password}@\${app-dns}**



NOTE

For more information on how to edit the Fabric8 profile, see [Fabric Guide](#)

Deploying the cartridge in a high availability environment

When you deploy this cartridge in an application cluster, note the following guidelines:

- The Fuse Builder cartridge supports auto-scaling. When you deploy this cartridge in an application cluster, specify auto-scaling to a minimum of 3 gears.
- In case of node failure, you must manually change the Jolokia URL in the **\fuse-builder\etc\settings.xml** file to connect to the active node. You specify the URL in the following property:

```
<fabric8.jolokiaUrl>http://[host_name]:8181/jolokia</fabric8.jolokiaUrl>
```

1.6. FABRIC MANAGEMENT

The JBoss A-MQ cartridge deploys and runs applications in a fabric. The fabric runs in the OpenShift Enterprise domain and manages all the applications that you create in that domain.

Make sure to note the following guidelines when you manage applications with fabric in an OpenShift Enterprise domain:

Child containers are not supported

When you create child containers in JBoss A-MQ, the containers inherit ports from the parent containers. However, OpenShift Enterprise applications must bind to specific ports. Therefore, you cannot create child containers in the EJBoss A-MQ on OpenShift Enterprise.

ZooKeeper server runs only on the first JBoss A-MQ application

The first JBoss A-MQ application that you create contains the ZooKeeper server instance with which each subsequent application authenticates. The ZooKeeper instance includes the user credentials and the environment variables required to run the applications in a fabric.

When you create subsequent applications, make sure to note the following guidelines:

- The primary application in the domain must be running when you create or start subsequent applications.

- The ZooKeeper credentials must be identical in all JBoss A-MQ instances that run in the same domain. You define the ZooKeeper password in the **OPENSIFT_FUSE_ZOOKEEPER_PASSWORD** property of the cartridge.
- If you want to delete the primary application, you must delete all the subsequent applications first.

UDP connections are not supported

Some Camel components require UDP network traffic routing. However, UDP is not supported in the JBoss A-MQ cartridge.

Secured shared file systems are not supported

Normally, you can restrict access to shared file systems such as NFS based on user ID. However, when you run JBoss A-MQ on OpenShift Enterprise, the user ID is dynamically generated at run-time. Therefore, you cannot configure the shared file system to restrict access based on user ID.

Some profiles are not supported

The following profiles are not supported when you run JBoss A-MQ on OpenShift Enterprise:

- controller-jon-server/
- controller-rhq-agent/
- controller-tomcat/
- controller-wildfly
- docker
- gateway-haproxy
- gateway-http
- gateway-mq
- hadoop-base
- hadoop-datanode
- hadoop-namenode
- jboss-brms-controller-tomcat
- jboss-brms-controller-wildfly
- jboss-brms-feature-workbench
- jboss-brms-feature-workbench.openshift
- openshift-aerogear-pushserver
- openshift-jbossews.1
- openshift-jbossews.2

In addition to the unsupported profiles, it is recommended not to deploy non-ActiveMQ profiles in your JBoss A-MQ application.

The OpenShift Enterprise Git repository is not used by the JBoss A-MQ cartridge

Normally, when you deploy an application, OpenShift Enterprise creates a Git repository with the source code of the application. However, JBoss A-MQ with fabric creates a standalone Git repository that stores all of the profiles and configuration files. Therefore, the cartridge does not use the Git repository that OpenShift Enterprise creates for the application.

1.7. HIGH AVAILABILITY

This section describes how to configure a JBoss A-MQ application cluster in a single OpenShift Enterprise domain.

How clustering works in the OpenShift Enterprise domain

When you deploy multiple JBoss A-MQ instances in a single OpenShift Enterprise domain, the first application that you create acts as the master node of the cluster. Each subsequent application that you create acts as a slave node.

Gear profile configuration

To prevent the master and slave applications from deploying on the same OpenShift Enterprise node, you must assign different a different gear profile to each set of nodes on which you deploy the JBoss A-MQ applications.

The gear profile properties do not need to be unique. You can assign the same gear profile to multiple master nodes or multiple slave nodes.

ZooKeeper ensemble server requirements

JBoss A-MQ supports management of multiple applications in an ensemble. However, when you deploy the JBoss A-MQ applications in OpenShift Enterprise, the ZooKeeper server runs inside the master application. Therefore, you cannot create ensembles of multiple applications.

To manage multiple applications in an ensemble, you must deploy a standalone JBoss A-MQ application and create an external ZooKeeper ensemble to manage the JBoss A-MQ instances that run in the OpenShift Enterprise domain.



NOTE

For more information, see [Fabric Guide](#)

Data storage and management

The JBoss A-MQ application cluster must use JDBC Master Slave to store and manage the data in the cluster.

Auto-scaling

OpenShift Enterprise supports auto-scaling of applications based on HTTP traffic. However, the JBoss A-MQ cartridge does not use HTTP to process data. Therefore, you cannot configure auto-scaling for the JBoss A-MQ cartridge.

1.8. UPGRADING THE JBOSS A-MQ CARTRIDGE

You upgrade the JBoss A-MQ cartridge in the same way that you apply asynchronous updates to other OpenShift Enterprise components. The JBoss A-MQ RPM package includes a ZIP package with the upgraded components.

For general information about OpenShift upgrades, see the [Asynchronous Errata Updates](#) section of the OpenShift Enterprise documentation.

1. Install the RPM package on every node in the OpenShift Enterprise domain with the following command:

```
yum update openshift-origin-cartridge-amq
```

2. Restart each node with the following command:

```
service ruby193-mcollective restart
```

3. On the OpenShift Enterprise broker, import the upgraded cartridge with the following commands:

```
oo-admin-ctl-cartridge -c import-profile --activate  
oo-admin-ctl-cartridge -c migrate
```

4. On each xPaaS node that runs JBoss A-MQ applications, upgrade the applications with the following commands:

```
oo-admin-upgrade upgrade-node --version <OSE_version_number>
```

The upgrade process applies the updated bundles to the relevant profiles and restarts all of the containers that run in the fabric.