Red Hat AMQ Broker 7.12

Release Notes for Red Hat AMQ Broker 7.12

Release Notes for AMQ Broker

Last Updated: 2024-05-21

Red Hat AMQ Broker 7.12 Release Notes for Red Hat AMQ Broker 7.12

Release Notes for AMQ Broker

Abstract

These release notes contain the latest information about new features, enhancements, fixes, and issues contained in the AMQ Broker 7.12 release.

Table of Contents

MAKING OPEN SOURCE MORE INCLUSIVE

Red Hat is committed to replacing problematic language in our code, documentation, and web properties. We are beginning with these four terms: master, slave, blacklist, and whitelist. Because of the enormity of this endeavor, these changes will be implemented gradually over several upcoming releases. For more details, see our CTO Chris Wright’s message .

CHAPTER 1. LONG TERM SUPPORT FOR AMQ BROKER 7.12

AMQ Broker 7.12 has been designated as a Long Term Support (LTS) release version. For details of the terms of an LTS release, see How long are AMQ LTS releases supported?

Support for Red Hat Enterprise Linux and OpenShift Container Platform

The AMQ Broker 7.12 LTS version supports:

- Red Hat Enterprise Linux 7, 8 and 9
- OpenShift Container Platform 4.12, 4.13, 4.14 or 4.15

Red Hat strives to ensure that AMQ Broker remains compatible with future versions of OpenShift Container Platform; however this compatibility cannot be guaranteed. Interoperability testing is performed for each new OpenShift Container Platform version. If no compatibility issues are found, the new OpenShift Container Platform version is added to the Red Hat AMQ Broker 7 Supported Configurations.

CHAPTER 2. SUPPORTED CONFIGURATIONS

For information on supported configurations, see Red Hat AMQ Broker 7 Supported Configurations .

Minimum Java version

At a minimum, AMQ Broker 7.12 requires Java version 11 to run.

Openwire support

AMQ 7 Broker has provided support for the Openwire protocol since its release in 2017 as a means to migrate client applications to AMQ 7. With the release of AMQ Broker 7.9.0 in 2021, the Openwire protocol was deprecated and customers were encouraged to migrate their existing Openwire client applications to one of the fully supported protocols of AMQ 7 (CORE, AMQP, MQTT, or STOMP). Starting with the AMQ Broker 8.0 release, the Openwire protocol will be removed from AMQ Broker.

CHAPTER 3. NEW AND CHANGED FEATURES

This section describes a highlighted set of enhancements and changed features in AMQ Broker 7.12.

AMQ Broker integration with cert-manager Operator for Openshift for certificate management

In AMQ Broker 7.12 on OpenShift, you can use cert-manager Operator for Openshift to create and manage the certificates required to configure TLS on AMQ Broker. For more information, see Using cert-manager Operator for Openshift in *Deploying AMQ Broker on Openshift*.

AMQ Broker Openshift service serving TLS certificate

If you want to secure internal connections between the broker and clients on the same Openshift cluster, you can add an annotation to the acceptor service to request that Openshift generates a service serving TLS certificate. For more information, see Using Openshift service serving certificates in *Deploying AMQ Broker on Openshift*.

Support for Privacy Enhanced Mail (PEM) certificates

In AMQ Broker 7.12 support for TLS certificates in PEM format is added.

Support for deployment in Openshift namespaces that have a restricted policy

You can deploy AMQ Broker 7.12 on OpenShift in namespaces that have a restricted OpenShift security context constraint, by default. In addition to the pod security options, you can customize the container security options in the CR if you want to run the broker in a different OpenShift security context. For more information, see Custom Resource configuration reference in *Deploying AMQ Broker on Openshift*.

Segregating brokerProperties configuration

If your custom resource (CR) for AMQ Broker 7.12 on OpenShift contains a **brokerProperties** section and the CR is at the maximum size limit of 1 MB, you can segregate the **brokerProperties** configuration into one or more Java properties files which you then reference in the CR. You might want to also segregate the **brokerProperties** configuration in separate files to logically group the **brokerProperties** items for easier maintenance. For more information, see Segregating the **brokerProperties** configuration in *Deploying AMQ Broker on Openshift*.

Support for ingresses in addition to routes on Openshift

In AMQ Broker 7.12 on OpenShift, you can expose acceptors, connectors and the management console to clients that are outside the Openshift cluster by using ingresses in addition to routes. For more information, see Configuring acceptors in *Deploying AMQ Broker on Openshift*.

Support for mounting shared volumes for third-party JAR files

In AMQ Broker 7.12 on OpenShift, you can configure the Operator to mount a shared volumes on each broker pod in a cluster. A use case for mounting a shared volume on each pod is to store a third-party JAR file, such as a JAR file for a JDBC database, required by the broker. On both RHEL and Openshift platforms, you can extend the Java classpath to make additional JAR files available to the broker at runtime. For more information, see Adding third-party JAR files in *Deploying AMQ Broker on Openshift*.

Customizing Openshift resources created by the Operator

In AMQ Broker 7.12 on OpenShift, you can customize Openshift resources such as deployments, pods and services that are created and managed by the Operator. Customizing these resources can be useful if you want to perform certain tasks such as:

- Adding custom annotations that control how resources are treated by other services.
- Modifying attributes that are not exposed in the broker CR.
  For more information, see Customizing Openshift resources created by the Operator in *Deploying AMQ Broker on Openshift*.

Support for adding plugins to AMQ Broker on Openshift

In AMQ Broker 7.12 on OpenShift, you can extend the functionality of AMQ Broker by registering plugins in the CR. For more information, see Registering plugins with AMQ Broker in *Deploying AMQ Broker on Openshift*.

Secure cluster connections support

In AMQ Broker 7.12 on Openshift, you can secure cluster connections by enabling SSL for internal acceptors and connectors. For more information, see Securing cluster connections in *Deploying AMQ Broker on Openshift*.

Automatic reloads of SSL artifacts

In AMQ Broker 7.12 on OpenShift and RHEL, you can configure AMQ Broker to reload renewed TLS certificates and other changes to the keystore or truststore configurations without requiring a broker restart. To configure automatic reload, set the sslAutoReload` attribute for the acceptor. For an example of how to configure automatic reloads of SSL artifacts on Openshift, see Using cert-manager Operator for Openshift in *Deploying AMQ Broker on Openshift*.

Health check for clustered brokers

In AMQ Broker 7.12, you can use the **artemis check cluster** command-line utility to verify the topology of the broker nodes in a cluster. For more information, see Checking the health of brokers, queues and clusters in *Configuring AMQ Broker*.

Federation support using AMQP broker connections

In AMQ Broker 7.12, you can configure federation of addresses and queues over an outbound AMQP broker connection. Using the AMQP protocol for federation offers the following benefits over using the Core protocol:

- If clients use the AMQP protocol for messaging, use the AMQP protocol for federation to eliminate the conversion of messages between AMQP and Core.
- AMQP federation supports two-way federation over a single outgoing connection. The two-way support eliminates the need for a remote broker to connect back to a local broker, which is a requirement when you use the Core protocol for federation and which might be prevented by network policies.
- AMQP federation provides better control over the movement of messages between brokers to avoid messages moving back and forth between brokers.

For more information, see Configuring federation using the AMQP protocol in *Configuring AMQ Broker*.

Use a custom shell from the command-line interface

In AMQ Broker 7.12, you can interact with the broker by using a custom **artemis** shell from the AMQ Broker command-line interface. The custom shell has built-in auto-completion of commands and command parameters. For more information, see Using the CLI in an **artemis** shell in *Managing AMQ Broker*.

Literal matching of addresses that contain wildcards

In AMQ Broker 7.12, you can configure literal matching to treat wildcard characters as literal characters for matching addresses that contain wildcards. For more information, see Configuring a literal match in *Configuring AMQ Broker*.

Role-based access control for JMX Management Operations

In AMQ Broker 7.12, you can use two new permissions, **view** and **edit**, to configure role-based access control for JMX Management Operations, without requiring a broker restart. For information on configuring role-based access control on RHEL, see Configuring role-based access control in the

broker.xml file in *Configuring AMQ Broker*. For information on configuring role-based access control on Openshift, see Configuring role-based access control for management operations in *Deploying AMQ Broker on Openshift*.

Change to the format of the output of the queue stat command

The format of the output of the **queue stat** command has changed from the output in 7.11 and earlier versions of AMQ Broker, which might affect automated processes that run in your deployment.

New parameter configurable in an MQTT acceptor to automatically remove MQTT subscription queues

In AMQ Broker 7.12, you can configure a **defaultMqttSessionExpiryInterval** parameter in an MQTT acceptor to automatically remove MQTT subscription queues that are not removed when the corresponding client session expires. The new parameter represents the number of **seconds** that must elapse after the client has disconnected before the broker removes the session state and subscription queues. Prior to 7.12, it was necessary to configure **auto-delete-*** parameters in an **address-setting** to remove a queue that was not removed when the client session expired.

Operator channels

The AMQ Broker Operator, **Red Hat Integration - AMQ Broker for RHEL 8 (Multiarch)**, is available with the following channels:

- **7.12.x** - This channel provides updates for version 7.12 only and is a Long Term Support (LTS) channel.
- **7.11.x** - This channel provides updates for version 7.11 only and is a Long Term Support (LTS) channel.
- **7.10.x** - This channel provides updates for version 7.10 only and is a Long Term Support (LTS) channel.

NOTE

It is not possible to upgrade the Operator by switching channels. You must uninstall the existing Operator and install the new version of the Operator from the appropriate channel.

To determine which Operator to choose, see the Red Hat Enterprise Linux Container Compatibility Matrix.

CHAPTER 4. DEPRECATED FEATURES

This section describes features that are supported, but have been deprecated from AMQ Broker.

ActiveMQArtemisAddress CRD

Starting in 7.12, the **ActiveMQArtemisAddress** CRD is deprecated. Use the **spec.brokerProperties** attribute in the **ActiveMQArtemis** CR to create addresses and queues for your deployment.

ActiveMQArtemisSecurity CRD

Starting in 7.12, the **ActiveMQArtemisSecurity** CRD is deprecated. Use the **spec.brokerProperties** attribute in the **ActiveMQArtemis** CR to configure security for your deployment.

ActiveMQArtemisScaledown CRD

Starting in 7.12, the **ActiveMQArtemisScaledown** CRD is deprecated. The **ActiveMQArtemisScaledown** CRD is used internally by the broker, so this change is transparent to AMQ Broker administrators.

Connection pooling for LDAP queries

Starting in 7.12, the **connectionPool** parameter, which enables connection pooling for LDAP queries, is deprecated. The built-in authorization and authentication caches provide an alternative way to optimize the performance of LDAP queries. For information on customizing the built-in caches, see Configuring authentication and authorization caching.

upgrade attribute in Custom Resource

Starting in 7.11, the **upgrade** attribute and the associated **enabled** and **minor** attributes are deprecated because they cannot work as originally designed. Use the **image** or **version** attributes to deploy specific broker container images.

queues configuration element

Starting in 7.10, the <queues> configuration element is deprecated. You can use the <addresses> configuration element to create addresses and associated queues. The <queues> configuration element will be removed in a future release.

getAddressesSettings method

Starting in 7.10, the get**Addresses**Settings method, which is included in the org.apache.activemq.artemis.core.config.Configuration interface, is deprecated. Use the get**Address**Settings method to configure addresses and queues for the broker programatically.

OpenWire protocol

Starting in 7.9, the OpenWire protocol is a deprecated feature. If you are creating a new AMQ Broker-based system, use one of the other supported protocols. Starting with the 8.0 release, the Openwire protocol will be removed from AMQ Broker.

Adding users when broker instance is not running

Starting in 7.8, when an AMQ Broker instance is not running, the ability to add users to the broker from the CLI interface is removed.

Network pinger

Starting in 7.5, network pinging is a deprecated feature. Network pinging cannot protect a broker cluster from network isolation issues that can lead to irrecoverable message loss. This feature will be removed in a future release. Red Hat continues to support existing AMQ Broker deployments that use network pinging. However, Red Hat no longer recommends use of network pinging in new deployments. For guidance on configuring a broker cluster for high availability and to avoid network isolation issues, see Implementing high availability in *Configuring AMQ Broker*.

Hawtio dispatch console plugin

Starting in 7.3, AMQ Broker no longer ships with the Hawtio dispatch console plugin, **dispatch-hawtio-console.war**. Previously, the dispatch console was used to manage AMQ Interconnect. However, AMQ Interconnect now uses its own, standalone web console.

CHAPTER 5. FIXED ISSUES

For a complete list of issues that have been fixed in the release, see AMQ Broker 7.12.0 Fixed Issues

CHAPTER 6. FIXED COMMON VULNERABILITIES AND EXPOSURES

This section details Common Vulnerabilities and Exposures (CVEs) fixed in the AMQ Broker 7.12 release.

- ENTMQBR-8644 - TRIAGE CVE-2023-6717 keycloak: XSS via assertion consumer service URL in SAML POST-binding flow [amq-7]
- ENTMQBR-8976 - TRIAGE CVE-2024-29025 netty-codec-http: Allocation of Resources Without Limits or Throttling [amq-7]
- ENTMQBR-8927 - CVE-2024-22259 springframework: URL Parsing with Host Validation [amq-7]
- ENTMQBR-8740 - CVE-2024-1132 keycloak: path transversal in redirection validation [amq-7]
- ENTMQBR-8758 - CVE-2024-1249 keycloak: org.keycloak.protocol.oidc: unvalidated cross-origin messages in checkLoginIframe leads to DDoS [amq-7]
- ENTMQBR-8626 - CVE-2023-6378 logback: serialization vulnerability in logback receiver [amq-7]
- ENTMQBR-8627 - CVE-2023-6481 logback: A serialization vulnerability in logback receiver [amq-7]
- ENTMQBR-8953 - CVE-2024-29131 CVE-2024-29133 commons-configuration2: various flaws [amq-7]
- ENTMQBR-8702 - CVE-2023-44981 zookeeper: Authorization Bypass in Apache ZooKeeper [amq-7]
- ENTMQBR-8611 - CVE-2022-41678 activemq: Apache ActiveMQ: Deserialization vulnerability on Jolokia that allows authenticated users to perform RCE [amq-7]
- ENTMQBR-8225 - CVE-2023-24540 amq-broker-rhel8-operator-container: golang: html/template: improper handling of JavaScript whitespace [amq-7]
- ENTMQBR-8227 - CVE-2022-21698 amq-broker-rhel8-operator-container: prometheus/client_golang: Denial of service using InstrumentHandlerCounter [amq-7]
- ENTMQBR-8238 - CVE-2022-21698 CVE-2023-24534 amq-broker-rhel8-operator-container: golang: net/http, net/textproto: denial of service from excessive memory allocation [amq-7]
- ENTMQBR-8239 - CVE-2023-29400 amq-broker-rhel8-operator-container: golang: html/template: improper handling of empty HTML attributes [amq-7]
- ENTMQBR-8240 - CVE-2023-24539 amq-broker-rhel8-operator-container: golang: html/template: improper sanitization of CSS values [amq-7]
- ENTMQBR-8228 - CVE-2021-43565 amq-broker-rhel8-operator-container: golang.org/x/crypto: empty plaintext packet causes panic [amq-7]
- ENTMQBR-8230 - CVE-2022-41723 amq-broker-rhel8-operator-container: net/http, golang.org/x/net/http2: avoid quadratic complexity in HPACK decoding [amq-7]

- ENTMQBR-8236 - CVE-2023-24536 amq-broker-rhel8-operator-container: golang: net/http, net/textproto, mime/multipart: denial of service from excessive resource consumption [amq-7]
- ENTMQBR-8237 - CVE-2023-24537 amq-broker-rhel8-operator-container: golang: go/parser: Infinite loop in parsing [amq-7]
- ENTMQBR-8231 - CVE-2022-2879 amq-broker-rhel8-operator-container: golang: archive/tar: unbounded memory consumption when reading headers [amq-7]
- ENTMQBR-8229 - CVE-2022-27664 amq-broker-rhel8-operator-container: golang: net/http: handle server errors after sending GOAWAY [amq-7]
- ENTMQBR-8226 - CVE-2022-32189 amq-broker-rhel8-operator-container: golang: math/big: decoding big.Float and big.Rat types can panic if the encoded message is too short, potentially allowing a denial of service [amq-7]
- ENTMQBR-8232 - CVE-2022-41715 amq-broker-rhel8-operator-container: golang: regexp/syntax: limit memory used by parsing regexps [amq-7]
- ENTMQBR-8241 - CVE-2023-24538 amq-broker-rhel8-operator-container: golang: html/template: backticks not treated as string delimiters [amq-7]
- ENTMQBR-8233 - CVE-2022-2880 amq-broker-rhel8-operator-container: golang: net/http/httputil: ReverseProxy should not forward unparseable query parameters [amq-7]
- ENTMQBR-8234 - CVE-2022-41724 amq-broker-rhel8-operator-container: golang: crypto/tls: large handshake records may cause panics [amq-7]
- ENTMQBR-8608 - CVE-2022-41678 activemq-broker-operator: Apache ActiveMQ: Deserialization vulnerability on Jolokia that allows authenticated users to perform RCE [amq-7]
- ENTMQBR-8235 - CVE-2022-41725 amq-broker-rhel8-operator-container: golang: net/http, mime/multipart: denial of service from excessive resource consumption [amq-7]
- ENTMQBR-8671 - CVE-2023-51074 json-path: stack-based buffer overflow in Criteria.parse method [amq-7]

CHAPTER 7. KNOWN ISSUES

This section describes known issues in AMQ Broker 7.12.

- ENTMQBR-9103 - **NullPointerException when closing multiple threads consuming AMQP**
  When running a multi-threaded consumer for AMQP messages, the broker sometimes generates WARN-level log messages similar to the following:

  ```
  2024-05-13 18:11:46,048 WARN [io.netty.util.concurrent.AbstractEventExecutor] A task raised
  an exception. Task:
  org.apache.activemq.artemis.protocol.amqp.proton.AMQPLargeMessageWriter$$Lambda$643/0
  java.lang.NullPointerException: null
  ```

  The messages are produced when the client finishes consuming messages and are sometimes accompanied by a stack trace.

  No messages are lost and the message can be ignored.

- ENTMQBR-8106 - **AMQ Broker Drainer pod doesn't function properly after changing MessageMigration in CR**
  You cannot change the value of the **messageMigration** attribute in a running broker deployment. To work around this issue, you must set the required value for the **messageMigration** attribute in a new **ActiveMQ Artemis** CR and create a new broker deployment.

- ENTMQBR-8166 - **Self-signed certificate with UseClientAuth=true prevents communication of Operator with Jolokia**
  If the **useClientAuth** attribute is set to **true** in the **console** section of the **ActiveMQ Artemis** CR, the Operator is unable to configure certain features, for example, create addresses, on the broker. In the Operator log, you see an error message that ends with **remote error: tls: bad certificate**.

- ENTMQBR-7359 - **Change to current handling of credential secret with 7.10.0 Operator**
  The Operator stores the administrator username and password for connecting to the broker in a secret. The default secret name is in the form ***<custom-resource-name>*-credentials-secret**. You can create a secret manually or allow the Operator to create a secret.

  If the **adminUser** and **adminPassword** attributes are configured in a Custom Resource prior to 7.10.0, the Operator updates a manually-created secret with the values of these attributes. Starting in 7.10.0, the Operator no longer updates a secret that was created manually. Therefore, if you change the values of the **adminUser** and **adminPassword** attributes in the CR, you must either:

  - Update the secret with the new username and password
  - Delete the secret and allow the Operator to create a secret. When the Operator creates a secret, it adds the values of the **adminUser** and **adminPassword** attributes if these are specified in the CR. If these attributes are not in the CR, the Operator generates random credentials for the secret.

- ENTMQBR-7111 - **7.10 versions of operator tend to remove StatefulSet during upgrade**
  If you are upgrading to or from AMQ Broker Operator 7.10.0, the new Operator automatically deletes the existing StatefulSet for each deployment during the reconciliation process. When the Operator deletes the StatefulSet, the existing broker pods are deleted, which causes a temporary broker outage.

You can work around this issue by running the following command to manually delete the StatefulSet and orphan the running pods before the Operator gets to delete the StatefulSet: oc delete statefulset *<statefulset-name>* --cascade=orphan

Manually deleting the StatefulSet during the upgrade process allows the new Operator to reconcile the StatefulSet without deleting the running pods. For more information, see Upgrading the Operator using OperatorHub in *Deploying AMQ Broker on OpenShift*.

- **ENTMQBR-5749 - Remove unsupported operators that are visible in OperatorHub**
  Only the Operators and Operator channels mentioned in Deploying the Operator from OperatorHub are supported. For technical reasons associated with Operator publication, other Operator and channels are visible in the OperatorHub and should be ignored. For reference, the following list shows which Operators are visible, but not supported:
  - Red Hat Integration - AMQ Broker LTS - all channels
  - Red Hat Integration - AMQ Broker - alpha, current, and current-76
- **ENTMQBR-4140 - AMQ Broker Operator: Installation becomes unusable if storage.size is improperly specified**
  If you configure the **storage.size** property of a Custom Resource (CR) instance to specify the size of the Persistent Volume Claim (PVC) required by brokers in a deployment for persistent storage, the Operator installation becomes unusable if you do not specify this value properly. For example, suppose that you set the value of **storage.size** to **1** (that is, without specifying a unit). In this case, the Operator cannot use the CR to create a broker deployment. In addition, even if you remove the CR and deploy a new version with **storage.size** specified correctly, the Operator still cannot use this CR to create a deployment as expected.

  To work around this issue, first stop the Operator. In the OpenShift Container Platform web console, click **Deployments**. For the Pod that corresponds to the AMQ Broker Operator, click the **More options** menu (three vertical dots). Click **Edit Pod Count** and set the value to **0**. When the Operator Pod has stopped, create a new version of the CR with **storage.size** correctly specified. Then, to restart the Operator, click **Edit Pod Count** again and set the value back to **1**.
- **ENTMQBR-4141 - AMQ Broker Operator: Increasing Persistent Volume size requires manual involvement even after recreating Stateful Set**
  If you try to increase the size of the Persistent Volume Claim (PVC) required by brokers in a deployment for persistent storage, the change does not take effect without further manual steps. For example, suppose that you configure the **storage.size** property of a Custom Resource (CR) instance to specify an initial size for the PVC. If you modify the CR to specify a *different* value of **storage.size**, the existing brokers continue to use the original PVC size. This is the case even if you scale the deployment down to zero brokers and then back up to the original number. However, if you scale the size of the deployment up to add additional brokers, the new brokers use the new PVC size.

  To work around this issue, and ensure that all brokers in the deployment use the same PVC size, use the OpenShift Container Platform web console to expand the PVC size used by the deployment. In the console, click **Storage → Persistent Volume Claims**. Click your deployment. On the **Actions** drop-down menu in the top-right corner, select **Expand PVC** and enter a new value.

CHAPTER 8. IMPORTANT LINKS

- Red Hat AMQ Broker 7.11 Release Notes
- Red Hat AMQ Broker 7.10 Release Notes
- Red Hat AMQ Broker 7.9 Release Notes
- Red Hat AMQ Broker 7.8 Release Notes
- Red Hat AMQ Broker 7.7 Release Notes
- Red Hat AMQ Broker 7.6 Release Notes
- Red Hat AMQ Broker 7.1 to 7.5 Release Notes (aggregated)
- Red Hat AMQ 7 Supported Configurations
- Red Hat AMQ 7 Component Details

Revised on 2024-05-21 10:53:52 UTC