Red Hat AMQ 7.6

Release Notes for AMQ Streams 1.4 on RHEL

For use with AMQ Streams on Red Hat Enterprise Linux

Last Updated: 2020-04-20

Red Hat AMQ 7.6 Release Notes for AMQ Streams 1.4 on RHEL

For use with AMQ Streams on Red Hat Enterprise Linux

Legal Notice

Abstract

These release notes contain the latest information about new features, enhancements, fixes, and issues contained in the AMQ Streams 1.4 release.

Table of Contents

CHAPTER 1. FEATURES

The features added in this release, and that were not in previous releases of AMQ Streams, are outlined below.

1.1. KAFKA 2.4.0 SUPPORT

AMQ Streams now supports Apache Kafka version 2.4.0.

AMQ Streams uses Kafka 2.4.0. Only Kafka distributions built by Red Hat are supported.

For upgrade instructions, see AMQ Streams and Kafka upgrades .

Refer to the Kafka 2.3.0 and Kafka 2.4.0 Release Notes for additional information.

NOTE

Kafka 2.3.x is supported in AMQ Streams 1.4 only for upgrade purposes.

For more information on supported versions, see the Red Hat AMQ 7 Component Details Page on the Customer Portal.

Changes to the partition rebalance protocol in Kafka 2.4.0

Kafka 2.4.0 adds *incremental cooperative rebalancing* for consumers and Kafka Streams applications. This is an improved rebalance protocol for implementing *partition rebalances* according to a defined *rebalance strategy*. Using the new protocol, consumers keep their assigned partitions during a rebalance and only revoke them at the end of the process if required to achieve cluster balance. This reduces the unavailability of the consumer group or Kafka Streams application during a rebalance.

To take advantage of incremental cooperative rebalancing, you must upgrade consumers and Kafka Streams applications to use the new protocol instead of the old *eager rebalance protocol*.

See Upgrading consumers and Kafka Streams applications to cooperative rebalancing and Notable changes in 2.4.0 in the Apache Kafka documentation.

1.2. ZOOKEEPER 3.5.7

Kafka version 2.4.0 requires a new version of ZooKeeper, version 3.5.7. Therefore, the first step involved in upgrading to AMQ Streams 1.4 is to upgrade ZooKeeper to version 3.5.7, as described in AMQ Streams and Kafka upgrades.

Refer to the ZooKeeper 3.5.7 Release Notes for additional information. Note the changes to the configuration format.

Before you start the upgrade, be aware of the following information relating to the new version of ZooKeeper.

1.2.1. New configuration options and address format

ZooKeeper 3.5.7 adds the following configuration options:

- **reconfigEnabled** - Enables or disables dynamic reconfiguration. Must be enabled in order to add or remove servers to a ZooKeeper 3.5.7 cluster.

- **standaloneEnabled** - Enables or disables standalone mode, where ZooKeeper runs with only one server.

The format of ZooKeeper server addresses has changed. The client port is now specified in the server address instead of the static ZooKeeper configuration file. For example:

```
server.1=172.17.0.1:2888:3888:participant;172.17.0.1:2181
server.2=172.17.0.2:2888:3888:participant;172.17.0.2:2181
server.3=172.17.0.3:2888:3888:participant;172.17.0.3:2181
```

You must apply these changes to the ZooKeeper configuration file as part of the ZooKeeper 3.5.7 upgrade.

1.2.2. Four letter word commands

ZooKeeper four letter word commands must now be added to the allow list before they can be used.

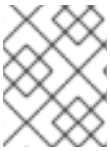
1.2.3. Missing snapshots error when upgrading to ZooKeeper 3.5.7

Snapshots check

To enable faster recovery from crashed nodes, at startup ZooKeeper 3.5.7 checks that at least one snapshot file exists in the data directory for the ZooKeeper cluster, **/var/lib/zookeeper/**. If the data directory does not contain any snapshot files, ZooKeeper 3.5.7 fails to start with the following error:

```
java.io.IOException: No snapshot found, but there are log entries. Something is broken!
```

If you see this error after upgrading ZooKeeper to version 3.5.7, you must perform some additional steps, described below. You must successfully complete the ZooKeeper 3.5.7 upgrade before proceeding with upgrading the Kafka brokers.



NOTE

Snapshot files are likely to exist for a long-running ZooKeeper cluster, so the missing snapshots error does not occur.

Additional steps to recover from missing snapshots error in ZooKeeper 3.5.7 upgrades

Perform the following steps if you see the missing snapshots error when performing the Upgrading ZooKeeper procedure.

1. Enable the **snapshot.trust.empty** configuration option in the ZooKeeper configuration file, **/opt/kafka/config/zookeeper.properties**. For example:

```
timeTick=2000
dataDir=/var/lib/zookeeper/
clientPort=2181
snapshot.trust.empty=true
```

2. Restart ZooKeeper:

```
/opt/kafka/bin/zookeeper-server-start.sh -daemon
/opt/kafka/config/zookeeper.properties
```

3. Wait until at least one snapshot file is created in the **/var/lib/zookeeper/** directory.

4. Disable the **snapshot.trust.empty** configuration option. For example:

```
timeTick=2000
dataDir=/var/lib/zookeeper/
clientPort=2181
snapshot.trust.empty=false
```

5. Restart ZooKeeper again:

```
/opt/kafka/bin/zookeeper-server-start.sh -daemon
/opt/kafka/config/zookeeper.properties
```

6. If the upgrade to ZooKeeper 3.5.7 completes without any errors, you can continue with the AMQ Streams 1.4 upgrade.

1.2.4. Scaling ZooKeeper 3.5.7 up or down

The configuration procedure for ZooKeeper 3.5.7 servers is significantly different than for ZooKeeper 3.4.x servers. Referred to as dynamic reconfiguration, the new configuration procedure requires that servers are added or removed using the ZooKeeper CLI or Admin API. This ensures that a stable ZooKeeper cluster is maintained during the scale up or scale down process.

To scale a ZooKeeper 3.5.7 cluster up or down, you must perform the procedures described here. *Scaling ZooKeeper up* means adding servers to a ZooKeeper cluster. *Scaling ZooKeeper down* means removing servers from a ZooKeeper cluster.

Scaling up ZooKeeper in an AMQ Streams 1.4 cluster

This procedure assumes that authentication is enabled for the ZooKeeper cluster, and you can access the server using the authentication mechanism configured for the ZooKeeper cluster.

Perform the following steps for each ZooKeeper server, **one at a time**:

1. Add a server to the ZooKeeper cluster as described in Running a multi-node ZooKeeper cluster and start ZooKeeper.

2. Note the IP address and configured access ports of the new server.

3. Start a **zookeeper-shell** session for the server. Run the following command from a machine that has access to the cluster (this might be one of the ZooKeeper nodes or your local machine, if it has access).

```
su - kafka
/opt/kafka/bin/zookeeper-shell.sh <ip-address>:<zk-port>
```

4. In the shell session, with the ZooKeeper node running, enter the following line to add the new server to the quorum as a voting member:

```
reconfig -add server.<positive-id> = <address1>:<port1>:<port2>[:role];[<client-port-address>:]<client-port>
```

   For example:

```
reconfig -add server.4=172.17.0.4:2888:3888:participant;172.17.0.4:2181
```

Where **<positive-id>** is the new server ID **4**.

For the two ports, **<port1>** *2888* is for communication between ZooKeeper servers, and **<port2>** *3888* is for leader election.

The new configuration propagates to the other servers in the ZooKeeper cluster; the new server is now a full member of the quorum.

5. Repeat steps 1-4 for the other servers that you want to add.

Scaling down ZooKeeper in an AMQ Streams 1.4 cluster

This procedure assumes that authentication is enabled for the ZooKeeper cluster. Before proceeding, read the notes on "Removing servers" in the ZooKeeper documentation.

Perform the following steps for each ZooKeeper server, **one at a time**:

1. Log in to the **zookeeper-shell** on one of the servers that will be **retained** after the scale down (for example, server 1).

NOTE

Access the server using the authentication mechanism configured for the ZooKeeper cluster.

2. Remove a server, for example server 5.

```
reconfig -remove 5
```

3. Deactivate the server that you removed.

4. Repeat steps 1-3 to reduce the cluster size.

1.3. OAUTH 2.0 AUTHENTICATION

Support for OAuth 2.0 authentication moves from a Technology Preview to a generally available component of AMQ Streams.

AMQ Streams supports the use of OAuth 2.0 authentication using the *SASL OAUTHBEARER* mechanism. Using OAuth 2.0 token based authentication, application clients can access resources on application servers (called 'resource servers') without exposing account credentials. The client presents an access token as a means of authenticating, which application servers can also use to find more information about the level of access granted. The authorization server handles the granting of access and inquiries about access.

In the context of AMQ Streams:

- Kafka brokers act as resource servers
- Kafka clients act as resource clients

The brokers and clients communicate with the OAuth 2.0 authorization server, as necessary, to obtain or validate access tokens.

For a deployment of AMQ Streams, OAuth 2.0 integration provides:

- Server-side OAuth 2.0 support for Kafka brokers
- Client-side OAuth 2.0 support for Kafka MirrorMaker, Kafka Connect and the Kafka Bridge

Red Hat Single Sign-On integration

You can deploy Red Hat Single Sign-On as an authorization server and configure it for integration with AMQ Streams.

You can use Red Hat Single Sign-On to:

- Configure authentication for Kafka brokers
- Configure and authorize clients
- Configure users and roles
- Obtain access and refresh tokens

See Using OAuth 2.0 token-based authentication .

CHAPTER 2. ENHANCEMENTS

The enhancements added in this release are outlined below.

2.1. KAFKA 2.4.0 ENHANCEMENTS

For an overview of the enhancements introduced with Kafka 2.4.0, refer to the Kafka 2.4.0 Release Notes.

2.2. KAFKA BRIDGE NOW SUPPORTS DISTRIBUTED TRACING

Distributed tracing using Jaeger is now supported for the Kafka Bridge component of AMQ Streams on Red Hat Enterprise Linux.

The Kafka Bridge generates traces when it sends and receives messages to and from HTTP clients, and when HTTP clients send requests to the Kafka Bridge REST API to create a consumer, retrieve messages, and so on. You can view these traces in the Jaeger user interface.

To enable tracing for the Kafka Bridge, add the following property to the Kafka Bridge properties file:

```
bridge.tracing=jaeger
```

You can then deploy the Kafka Bridge to an AMQ Streams cluster on Red Hat Enterprise Linux by running the **bin/kafka_bridge_run.sh** script.

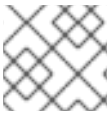
See Distributed tracing and Enabling tracing for the Kafka Bridge .

CHAPTER 3. TECHNOLOGY PREVIEWS

IMPORTANT

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. OAUTH 2.0 AUTHORIZATION



NOTE

This is a Technology Preview feature.

If you are using OAuth 2.0 for token-based authentication, you can now also use Keycloak to configure authorization rules to constrain client access to Kafka brokers.

Red Hat Single Sign-On 7.3 does not support this Technoloy Preview of OAuth 2.0 token-based authorization. If you wish to try this feature, it is tested for use in a development environment with Keycloak 8.0.2 as the authorization server.

AMQ Streams supports the use of OAuth 2.0 token-based authorization through Keycloak Authorization Services, which allows you to manage security policies and permissions centrally.

Security policies and permissions defined in Keycloak are used to grant access to resources on Kafka brokers. Users and clients are matched against policies that permit access to perform specific actions on Kafka brokers.

See Using OAuth 2.0 token-based authorization .

3.2. MIRRORMAKER 2.0

NOTE

This is a Technology Preview feature.

You can now use MirrorMaker 2.0 with AMQ Streams.

MirrorMaker 2.0 is based on the Kafka Connect framework, *connectors* managing the transfer of data between clusters.

MirrorMaker 2.0 uses:

- Source cluster configuration to consume data from the source cluster
- Target cluster configuration to output data to the target cluster

MirrorMaker 2.0 introduces an entirely new way of replicating data in clusters. If you choose to use MirrorMaker 2.0, there is currently no legacy support, so any resources must be manually converted into the new format.

See Using AMQ Streams with MirrorMaker 2.0 .

3.3. DEBEZIUM FOR CHANGE DATA CAPTURE INTEGRATION

Debezium for Change Data Capture is a distributed platform that monitors databases and creates change event streams. Debezium is built on Apache Kafka and can be deployed and integrated with AMQ Streams. Following a deployment of AMQ Streams, you deploy Debezium as a connector configuration through Kafka Connect. Debezium captures row-level changes to a database table and passes corresponding change events to AMQ Streams on Red Hat Enterprise Linux. Applications can read these *change event streams* and access the change events in the order in which they occurred.

Debezium has multiple uses, including:

- Data replication
- Updating caches and search indexes
- Simplifying monolithic applications
- Data integration
- Enabling streaming queries

Debezium provides connectors (based on Kafka Connect) for the following common databases:

- MySQL
- PostgreSQL
- SQL Server
- MongoDB

For more information on deploying Debezium with AMQ Streams, refer to the product documentation.

CHAPTER 4. DEPRECATED FEATURES

There are no deprecated features for AMQ Streams 1.4.

CHAPTER 5. FIXED ISSUES

The following table lists the issues fixed in AMQ Streams 1.4.

Issue Number	Description
ENTMQST-1717	Fix build of ZSTD library

CHAPTER 6. KNOWN ISSUES

There are no known issues for AMQ Streams 1.4.

CHAPTER 7. SUPPORTED INTEGRATION PRODUCTS

AMQ Streams 1.4 supports integration with the following Red Hat products.

- **Red Hat Single Sign-On 7.3** for OAuth 2.0 authentication (and OAuth 2.0 authorization with Keycloak as a Technology Preview)
- **Red Hat Debezium 1.0 and later** for monitoring databases and creating event streams

For information on the functionality these products can introduce to your AMQ Streams deployment, refer to the AMQ Streams 1.4 documentation.

CHAPTER 8. IMPORTANT LINKS

- Red Hat AMQ 7 Supported Configurations
- Red Hat AMQ 7 Component Details

Revised on 2020-04-20 14:36:43 UTC