Red Hat AMQ Streams 2.5

Release Notes for AMQ Streams 2.5 on RHEL

Highlights of what's new and what's changed with this release of AMQ Streams on Red Hat Enterprise Linux

Last Updated: 2024-02-22

Red Hat AMQ Streams 2.5 Release Notes for AMQ Streams 2.5 on RHEL

Highlights of what's new and what's changed with this release of AMQ Streams on Red Hat Enterprise Linux

Legal Notice

Abstract

The release notes summarize the new features, enhancements, and fixes introduced in the AMQ Streams 2.5 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. AMQ STREAMS 2.5 LONG TERM SUPPORT

AMQ Streams 2.5 is a Long Term Support (LTS) offering for AMQ Streams.

For information on the LTS terms and dates, see the AMQ Streams LTS Support Policy .

CHAPTER 2. FEATURES

AMQ Streams 2.5 introduces the features described in this section.

AMQ Streams 2.5 on RHEL is based on Apache Kafka 3.5.0.

NOTE

To view all the enhancements and bugs that are resolved in this release, see the AMQ Streams Jira project.

2.1. AMQ STREAMS 2.5.X (LONG TERM SUPPORT)

AMQ Streams 2.5.x is the Long Term Support (LTS) offering for AMQ Streams.

The latest patch release is AMQ Streams 2.5.1. The AMQ Streams product images have changed to version 2.5.1. The supported Kafka version remains at 3.5.0.

For information on the LTS terms and dates, see the AMQ Streams LTS Support Policy .

2.2. KAFKA 3.5.0 SUPPORT

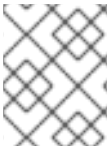
AMQ Streams now supports and uses Apache Kafka version 3.5.0. Only Kafka distributions built by Red Hat are supported.

For upgrade instructions, see AMQ Streams and Kafka upgrades .

Refer to the Kafka 3.5.0 Release Notes for additional information.

Kafka 3.4.x is supported only for the purpose of upgrading to AMQ Streams 2.5.

Kafka 3.5.0 uses ZooKeeper version 3.6.4, which is a different version to Kafka 3.4.x. We recommend that you perform a rolling update to use the new binaries.



NOTE

Kafka 3.5.0 provides access to KRaft mode, where Kafka runs without ZooKeeper by utilizing the Raft protocol. KRaft mode is available as a Technology Preview.

2.3. OPENTELEMETRY FOR DISTRIBUTED TRACING

OpenTelemetry for distributed tracing has moved to GA. You can use OpenTelemetry with a specified tracing system. OpenTelemetry has replaced OpenTracing for distributed tracing. Support for OpenTracing is deprecated.

By Default, OpenTelemetry uses the OTLP (OpenTelemetry Protocol) exporter for tracing. AMQ Streams with OpenTelemetry is distributed for use with the Jaeger exporter, but you can specify other tracing systems supported by OpenTelemetry. AMQ Streams plans to migrate to using OpenTelemetry with the OTLP exporter by default, and is phasing out support for the Jaeger exporter.

See Introducing distributed tracing.

CHAPTER 3. ENHANCEMENTS

AMQ Streams 2.5 adds a number of enhancements.

3.1. KAFKA 3.5.0 ENHANCEMENTS

For an overview of the enhancements introduced with Kafka 3.5.0, refer to the Kafka 3.5.0 Release Notes.

3.2. OAUTH 2.0 SUPPORT FOR KRAFT MODE

KeycloakRBACAuthorizer, the Red Hat Single Sign-On authorizer provided with AMQ Streams, has been replaced with the **KeycloakAuthorizer**. The new authorizer is compatible with using AMQ Streams with ZooKeeper cluster management or in KRaft mode. As with the previous authorizer, to be able to use the Red Hat Single Sign-On REST endpoints for Authorization Services provided by Red Hat Single Sign-On, you configure **KeycloakAuthorizer** on the Kafka broker. **KeycloakRBACAuthorizer** can still be used when using AMQ Streams with ZooKeeper cluster management, but you should migrate to the new authorizer.

3.3. OAUTH 2.0 CONFIGURATION PROPERTIES FOR GRANT MANAGEMENT

You can now use additional configuration to manage OAuth 2.0 grants from the authorization server.

If you are using Red Hat Single Sign-On for OAuth 2.0 authorization, you can add the following properties to the authorization configuration of your Kafka brokers:

- **strimzi.authorization.grants.max.idle.time.seconds** specifies the time in seconds after which an idle grant in the cache can be evicted. The default value is 300.
- **strimzi.authorization.grants.gc.period.seconds** specifies the time, in seconds, between consecutive runs of a job that cleans stale grants from the cache. The default value is 300.
- **strimzi.authorization.reuse.grants** controls whether the latest grants are fetched for a new session. When disabled, grants are retrieved from Red Hat Single Sign-On and cached for the user. The default value is **true**.

Kafka configuration to use OAuth 2.0 authorization

```
strimzi.authorization.grants.max.idle.time.seconds="300"
strimzi.authorization.grants.gc.period.seconds="300"
strimzi.authorization.reuse.grants="false"
```

See Configuring OAuth 2.0 authorization support.

3.4. OAUTH 2.0 SUPPORT FOR JSONPATH QUERIES WHEN EXTRACTING USERNAMES

To use OAuth 2.0 authentication in a Kafka cluster, you specify listener configuration with an OAUTH authentication mechanism. When configuring the listener properties, it is now possible to use a JsonPath query to extract a username from the authorization server being used. You can use a JsonPath query to specify username extraction options in your listener for the **oauth.username.claim**

and **oauth.fallback.username.claim** properties. This allows you to extract a username from a token by accessing a specific value within a nested data structure. For example, you might have a username that is contained within a *user info* data structure within a JSON token data structure.

The following example shows how JsonPath queries are specified for the properties when configuring token validation using an introspection endpoint.

Configuring token validation using an introspection endpoint

```
# ...
listener.name.client.oauthbearer.sasl.jaas.config=org.apache.kafka.common.security.oauthbearer.OAuthBearerLoginModule required ;
  # ...
  oauth.username.claim="['user.info'].['user.id']" \ (1)
  oauth.fallback.username.claim="['client.info'].['client.id']" \ (2)
  # ...
```

1. The token claim (or key) that contains the actual user name in the token. The user name is the *principal* used to identify the user. The **userNameClaim** value depends on the authorization server used.

2. An authorization server may not provide a single attribute to identify both regular users and clients. When a client authenticates in its own name, the server might provide a *client ID*. When a user authenticates using a username and password, to obtain a refresh token or an access token, the server might provide a *username* attribute in addition to a client ID. Use this fallback option to specify the username claim (attribute) to use if a primary user ID attribute is not available.

See Configuring OAuth 2.0 support for Kafka brokers .

3.5. KAFKA BRIDGE ENHANCEMENTS FOR METRICS AND OPENAPI

The latest release of the Kafka Bridge introduces the following changes:

- Removes the **remote** and **local** labels from HTTP server-related metrics to prevent time series sample growth.
- Eliminates accounting HTTP server metrics for requests on the /**metrics** endpoint.
- Exposes the /**metrics** endpoint through the OpenAPI specification, providing a standardized interface for metrics access and management.
- Fixes the **OffsetRecordSentList** component schema to return record offsets or errors.
- Fixes the **ConsumerRecord** component schema to return key and value as objects, not just (JSON) strings.
- Corrects the HTTP status codes returned by the /**ready** and /**healthy** endpoints:
  - Changes the successful response code from **200** to **204**, indicating no content in the response for success.
  - Adds the **500** status code to the specification for the failure case, indicating no content in the response for errors.

See Using the AMQ Streams Kafka Bridge .

CHAPTER 4. TECHNOLOGY PREVIEWS

Technology Preview features included with AMQ Streams 2.5.

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 the support scope, see Technology Preview Features Support Scope.

4.1. KRAFT MODE

Apache Kafka is in the process of phasing out the need for ZooKeeper. You can now try deploying a Kafka cluster in KRaft (Kafka Raft metadata) mode without ZooKeeper as a technology preview.

CAUTION

This mode is intended **only** for development and testing, and must not be enabled for a production environment.

Currently, the KRaft mode in AMQ Streams has the following major limitations:

- Moving from Kafka clusters with ZooKeeper to KRaft clusters or the other way around is not supported.
- Upgrades and downgrades of Apache Kafka versions are not supported.
- JBOD storage with multiple disks is not supported.
- Many configuration options are still in development.

See Running Kafka in KRaft mode .

4.2. KAFKA STATIC QUOTA PLUGIN CONFIGURATION

Use the technology preview of the *Kafka Static Quota* plugin to set throughput and storage limits on brokers in your Kafka cluster. You can set a byte-rate threshold and storage quotas to put limits on the clients interacting with your brokers.

Example Kafka Static Quota plugin configuration

```
client.quota.callback.class= io.strimzi.kafka.quotas.StaticQuotaCallback
client.quota.callback.static.produce= 1000000
client.quota.callback.static.fetch= 1000000
client.quota.callback.static.storage.soft= 400000000000
client.quota.callback.static.storage.hard= 500000000000
client.quota.callback.static.storage.check-interval= 5
```

See Setting limits on brokers using the Kafka Static Quota plugin .

CHAPTER 5. DEPRECATED FEATURES

The features deprecated in this release, and that were supported in previous releases of AMQ Streams, are outlined below.

5.1. JAVA 8 SUPPORT REMOVED IN AMQ STREAMS 2.4.0

Support for Java 8 was deprecated in Kafka 3.0.0 and AMQ Streams 2.0. Support for Java 8 was removed in AMQ Streams 2.4.0. This applies to all AMQ Streams components, including clients.

AMQ Streams supports Java 11 and Java 17. Use Java 11 or 17 when developing new applications. Plan to migrate any applications that currently use Java 8 to Java 11 or 17.

If you want to continue using Java 8 for the time being, AMQ Streams 2.2 provides Long Term Support (LTS). For information on the LTS terms and dates, see the AMQ Streams LTS Support Policy .

5.2. OPENTRACING

Support for OpenTracing is deprecated.

The Jaeger clients are now retired and the OpenTracing project archived. As such, we cannot guarantee their support for future Kafka versions. We are introducing a new tracing implementation based on the OpenTelemetry project.

5.3. KAFKA MIRRORMAKER 2 IDENTITY REPLICATION POLICY

Identity replication policy is a feature used with MirrorMaker 2 to override the automatic renaming of remote topics. Instead of prepending the name with the source cluster's name, the topic retains its original name. This setting is particularly useful for active/passive backups and data migration scenarios.

To implement an identity replication policy, you must specify a replication policy class (**replication.policy.class**) in the MirrorMaker 2 configuration. Previously, you could specify the **io.strimzi.kafka.connect.mirror.IdentityReplicationPolicy** class included with the AMQ Streams **mirror-maker-2-extensions** component. However, this component is now deprecated and will be removed in the future. Therefore, it is recommended to update your implementation to use Kafka's own replication policy class (**org.apache.kafka.connect.mirror.IdentityReplicationPolicy**).

See Using AMQ Streams with MirrorMaker 2 .

5.4. KAFKA MIRRORMAKER 1

Kafka MirrorMaker replicates data between two or more active Kafka clusters, within or across data centers. Kafka MirrorMaker 1 was deprecated in Kafka 3.0.0 and will be removed in Kafka 4.0.0. MirrorMaker 2 will be the only version available. MirrorMaker 2 is based on the Kafka Connect framework, connectors managing the transfer of data between clusters.

As a result, MirrorMaker 1 has also been deprecated in AMQ Streams as well. If you are using MirrorMaker 1 (referred to as just *MirrorMaker* in the AMQ Streams documentation), use MirrorMaker 2 with the **IdentityReplicationPolicy** class. MirrorMaker 2 renames topics replicated to a target cluster. **IdentityReplicationPolicy** configuration overrides the automatic renaming. Use it to produce the same active/passive unidirectional replication as MirrorMaker 1.

See Using AMQ Streams with MirrorMaker 2 .

CHAPTER 6. FIXED ISSUES

The following sections list the issues fixed in AMQ Streams 2.5.x. Red Hat recommends that you upgrade to the latest patch release.

For details of the issues fixed in Kafka 3.5.0, refer to the Kafka 3.5.0 Release Notes.

6.1. FIXED ISSUES FOR AMQ STREAMS 2.5.1

The AMQ Streams 2.5.1 patch release (Long Term Support) is now available.

KAFKA-15353

The 2.5.1 patch release includes a fix for KAFKA-15353, an issue that was included in the Kafka 3.5.2 release. Note that the patch release introduced a fix for this specific issue, not all issues fixed for Kafka 3.5.2.

For more information on the issue, see the Kafka 3.5.2 Release Notes.

HTTP/2 DoS vulnerability (CVE-2023-44487)

The release addresses CVE-2023-44487, a critical Denial of Service (DoS) vulnerability in the HTTP/2 protocol. The vulnerability stems from mishandling multiplexed streams, allowing a malicious client to repeatedly request new streams and promptly cancel them using an **RST_STREAM** frame. By doing so, the attacker forces the server to expend resources setting up and tearing down streams without reaching the server-side limit for active streams per connection. For more information on this vulnerability, see the CVE-2023-44487 page for a description.

For additional details about the issues resolved in AMQ Streams 2.5.1, see AMQ Streams 2.5.x Resolved Issues.

6.2. FIXED ISSUES FOR AMQ STREAMS 2.5.0

Table 6.1. Fixed issues

Issue Number	Description
ENTMQST-3757	[KAFKA] Mirror Maker 2 negative lag
ENTMQST-4496	[BRIDGE] Logged HTTP response status code could be different from the actual one returned to the client
ENTMQST-4707	Make connector task backoff configurable in Kafka Connect

Table 6.2. Fixed common vulnerabilities and exposures (CVEs)

Issue Number	Description
ENTMQST-4484	snakeyaml: Constructor Deserialization Remote Code Execution
ENTMQST-4995	TRIAGE-CVE-2023-34454 snappy-java-repolib: snappy-java: Integer overflow in compress leads to DoS

Issue Number	Description
ENTMQST-4996	TRIAGE-CVE-2023-34454 snappy-java-debuginfo: snappy-java: Integer overflow in compress leads to DoS
ENTMQST-4997	TRIAGE-CVE-2023-34454 snappy-java: Integer overflow in compress leads to DoS
ENTMQST-4998	TRIAGE-CVE-2023-34455 snappy-java: Unchecked chunk length leads to DoS
ENTMQST-5120	CVE-2023-34462 Flaw in Netty's SniHandler while navigating TLS handshake; DoS
ENTMQST-5121	CVE-2023-0482 RESTEasy: creation of insecure temp files
ENTMQST-5122	CVE-2022-24823 netty: world readable temporary file containing sensitive data
ENTMQST-5123	CVE-2021-37137 netty-codec: SnappyFrameDecoder doesn't restrict chunk length and may buffer skippable chunks in an unnecessary way
ENTMQST-5124	CVE-2021-37136 netty-codec: Bzip2Decoder doesn't allow setting size restrictions for decompressed data
ENTMQST-5125	CVE-2023-3635 DoS of the Okio client when handling a crafted GZIP archive
ENTMQST-5126	CVE-2023-26048 Jetty servlets with multipart support may cause OOM error with client requests
ENTMQST-5127	CVE-2023-26049 Non-standard cookie parsing in Jetty may allow an attacker to smuggle cookies within other cookies
ENTMQST-5128	CVE-2022-36944 scala: deserialization gadget chain
ENTMQST-5134	TRIAGE-CVE-2023-3635 okio: GzipSource class improper exception handling
ENTMQST-5178	CVE-2023-26048 jetty-server: OutOfMemoryError for large multipart without filename read via request.getParameter()
ENTMQST-5179	CVE-2023-26049 jetty-server: Cookie parsing of quoted values can exfiltrate values from other cookies

CHAPTER 7. KNOWN ISSUES

This section lists the known issues for AMQ Streams 2.5 on RHEL.

7.1. JMX AUTHENTICATION WHEN RUNNING IN FIPS MODE

When running AMQ Streams in FIPS mode with JMX authentication enabled, clients may fail authentication. To work around this issue, do not enable JMX authentication while running in FIPS mode. We are investigating the issue and working to resolve it in a future release.

CHAPTER 8. SUPPORTED CONFIGURATIONS

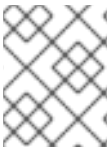
Supported configurations for the AMQ Streams 2.5 release.

8.1. SUPPORTED PLATFORMS

The following platforms are tested for AMQ Streams 2.5 running with Kafka on the version of Red Hat Enterprise Linux (RHEL) stated.

Operating System	Architecture	JVM
RHEL 7	x86, amd64	Java 11
RHEL 8 and 9	x86, amd64, ppc64le (IBM Power), s390x (IBM Z and IBM® LinuxONE), aarch64 (64-bit ARM)	Java 11 and Java 17

Platforms are tested with Open JDK 11 and 17. The IBM JDK is supported but not regularly tested against during each release. Open JDK 8, Oracle JDK 8 & 11, and IBM JDK 8 are not supported.



NOTE

Support for aarch64 (64-bit ARM) applies to AMQ Streams 2.5 when running Kafka 3.5.0 only.

8.2. SUPPORTED APACHE KAFKA ECOSYSTEM

In AMQ Streams, only the following components released directly from the Apache Software Foundation are supported:

- Apache Kafka Broker
- Apache Kafka Connect
- Apache MirrorMaker
- Apache MirrorMaker 2
- Apache Kafka Java Producer, Consumer, Management clients, and Kafka Streams
- Apache ZooKeeper

NOTE

Apache ZooKeeper is supported solely as an implementation detail of Apache Kafka and should not be modified for other purposes. Additionally, the cores or vCPU allocated to ZooKeeper nodes are not included in subscription compliance calculations. In other words, ZooKeeper nodes do not count towards a customer's subscription.

8.3. ADDITIONAL SUPPORTED FEATURES

- Kafka Bridge
- Drain Cleaner
- Cruise Control
- Distributed Tracing

See also, Chapter 10, *Supported integration with Red Hat products* .

8.4. STORAGE REQUIREMENTS

Kafka requires block storage; file storage options like NFS are not compatible.

Additional resources

For information on the supported configurations for the AMQ Streams 2.2 LTS release, see the AMQ Streams Supported Configurations article on the customer portal.

CHAPTER 9. COMPONENT DETAILS

The following table shows the component versions for each AMQ Streams release.

AMQ Streams	Apache Kafka	Strimzi Operators	Kafka Bridge	Oauth	Cruise Control
2.5.1	3.5.0	0.36.0	0.26	0.13.0	2.5.123
2.5.0	3.5.0	0.36.0	0.26	0.13.0	2.5.123
2.4.0	3.4.0	0.34.0	0.25.0	0.12.0	2.5.112
2.3.0	3.3.1	0.32.0	0.22.3	0.11.0	2.5.103
2.2.2	3.2.3	0.29.0	0.21.5	0.10.0	2.5.103
2.2.1	3.2.3	0.29.0	0.21.5	0.10.0	2.5.103
2.2.0	3.2.3	0.29.0	0.21.5	0.10.0	2.5.89
2.1.0	3.1.0	0.28.0	0.21.4	0.10.0	2.5.82
2.0.1	3.0.0	0.26.0	0.20.3	0.9.0	2.5.73
2.0.0	3.0.0	0.26.0	0.20.3	0.9.0	2.5.73
1.8.4	2.8.0	0.24.0	0.20.1	0.8.1	2.5.59
1.8.0	2.8.0	0.24.0	0.20.1	0.8.1	2.5.59
1.7.0	2.7.0	0.22.1	0.19.0	0.7.1	2.5.37
1.6.7	2.6.3	0.20.1	0.19.0	0.6.1	2.5.11
1.6.6	2.6.3	0.20.1	0.19.0	0.6.1	2.5.11
1.6.5	2.6.2	0.20.1	0.19.0	0.6.1	2.5.11
1.6.4	2.6.2	0.20.1	0.19.0	0.6.1	2.5.11
1.6.0	2.6.0	0.20.0	0.19.0	0.6.1	2.5.11
1.5.0	2.5.0	0.18.0	0.16.0	0.5.0	-
1.4.1	2.4.0	0.17.0	0.15.2	0.3.0	-

AMQ Streams	Apache Kafka	Strimzi Operators	Kafka Bridge	Oauth	Cruise Control
1.4.0	2.4.0	0.17.0	0.15.2	0.3.0	-
1.3.0	2.3.0	0.14.0	0.14.0	0.1.0	-
1.2.0	2.2.1	0.12.1	0.12.2	-	-
1.1.1	2.1.1	0.11.4	-	-	-
1.1.0	2.1.1	0.11.1	-	-	-
1.0	2.0.0	0.8.1	-	-	-

NOTE

Strimzi 0.26.0 contains a Log4j vulnerability. The version included in the product has been updated to depend on versions that do not contain the vulnerability.

CHAPTER 10. SUPPORTED INTEGRATION WITH RED HAT PRODUCTS

AMQ Streams 2.5 supports integration with the following Red Hat products:

Red Hat Single Sign-On

Provides OAuth 2.0 authentication and OAuth 2.0 authorization.

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

10.1. RED HAT SINGLE SIGN-ON

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

Additional resources

- Red Hat Single Sign-On Supported Configurations

Revised on 2024-02-22 14:12:28 UTC