Red Hat Integration 2023.q1

Release Notes for Red Hat Integration 2023.q1

What's new in Red Hat Integration

Last Updated: 2023-05-15

Red Hat Integration 2023.q1 Release Notes for Red Hat Integration 2023.q1

What's new in Red Hat Integration

Legal Notice

Abstract

Describes the Red Hat Integration product and provides the latest details on what's new in this release.

Table of Contents

CHAPTER 1. RED HAT INTEGRATION

Red Hat Integration is a comprehensive set of integration and event processing technologies for creating, extending, and deploying container-based integration services across hybrid and multicloud environments. Red Hat Integration provides an agile, distributed, and API-centric solution that organizations can use to connect and share data between applications and systems required in a digital world.

Red Hat Integration includes the following capabilities:

- Real-time messaging
- Cross-datacenter message streaming
- API connectivity
- Application connectors
- Enterprise integration patterns
- API management
- Data transformation
- Service composition and orchestration

Additional resources

- Understanding enterprise integration

CHAPTER 2. CAMEL EXTENSIONS FOR QUARKUS 2.13.2 RELEASE NOTES

2.1. CAMEL EXTENSIONS FOR QUARKUS FEATURES

Fast startup and low RSS memory

Using the optimized build-time and ahead-of-time (AOT) compilation features of Quarkus, your Camel application can be pre-configured at build time resulting in fast startup times.

Application generator

Use the Quarkus application generator to bootstrap your application and discover its extension ecosystem.

Highly configurable

All of the important aspects of a Camel Extensions for Quarkus application can be set up programmatically with CDI (Contexts and Dependency Injection) or via configuration properties. By default, a CamelContext is configured and automatically started for you.
Check out the Configuring your Quarkus applications guide for more information on the different ways to bootstrap and configure an application.

Integrates with existing Quarkus extensions

Camel Extensions for Quarkus provides extensions for libraries and frameworks that are used by some Camel components which inherit native support and configuration options.

2.2. SUPPORTED PLATFORMS, CONFIGURATIONS, DATABASES, AND EXTENSIONS

- For information about supported platforms, configurations, and databases in Camel Extensions for Quarkus version 2.13.2, see the Supported Configuration page on the Customer Portal (login required).

- For a list of Red Hat Camel Extensions for Quarkus extensions and the Red Hat support level for each extension, see the Extensions Overview chapter of the *Camel Extensions for Quarkus Reference* (login required).

- To configure your Red Hat Camel Extensions for Quarkus version 2.13.2 projects to use the supported extensions, use the latest Bill Of Materials (BOM) version 2.13.x version from https://maven.repository.redhat.com/ga/com/redhat/quarkus/platform/quarkus-bom. For more information about BOM dependency management, see Developing Applications with Camel Extensions for Quarkus

2.3. TECHNOLOGY PREVIEW EXTENSIONS

Red Hat does not provide support for Technology Preview components provided with this release of Camel Extensions for Quarkus. Items designated as Technology Preview in the Extensions Overview chapter of the *Camel Extensions for Quarkus Reference* have limited supportability, as defined by the Technology Preview Features Support Scope.

NOTE

CXF is a new Technology Preview component that is provided with this release of Camel Extensions for Quarkus. The following features are excluded from Technology Preview support:

- RESTful services
- CORBA support
- UDP Transport
- WS-MetadataExchange
- WS-Discovery

2.4. KNOWN CXF ISSUES

CEQ-5348 WS-SecurityPolicy, WS-Trust

When you use **WS-SecurityPolicy** or **WS-Trust** with a Camel CXF consumer, for example **from("cxf: ...")**, the SOAP endpoint throws the following exception:

```
SoapFault: BSP:R3227: A SECURITY_HEADER MUST NOT contain more than one TIMESTAMP
```

You can workaround this issue by deploying the SOAP endpoint using the same approach as the Quarkiverse CXF extension.

First, you require an implementation of your service endpoint interface (SEI) that forwards your message to a Camel route. For example, if your SEI is similar to the following:

```
@WebService(targetNamespace = "https://quarkiverse.github.io/quarkiverse-docs/quarkus-cxf/ws-securitypolicy")
public interface WssSecurityPolicyHelloService {
    @WebMethod
    String sayHello(String name);
}
```

Then, you must configure the implementation of your SEI as follows:

```
@WebService(portName = "EncryptSecurityServicePort", serviceName = "WssSecurityPolicyHelloService", targetNamespace = "https://quarkiverse.github.io/quarkiverse-docs/quarkus-cxf/ws-securitypolicy", endpointInterface = "org.apache.camel.quarkus.component.cxf.soap.securitypolicy.server.cxf.way.it.WssSecurityPolicyHelloService")
public class WssSecurityPolicyHelloServiceImpl implements WssSecurityPolicyHelloService {
    @Inject
    ProducerTemplate producerTemplate;

    public String sayHello(String name) {
        return producerTemplate.requestBody("direct:myDirectEndpoint", name, String.class);
    }
}
```

Next, in the **application.properties** file, you must configure the path that will serve the service. For example:

```
quarkus.cxf.path=/soap
quarkus.cxf.endpoint."/security-policy-
hello".implementor=org.apache.camel.quarkus.component.cxf.soap.securitypolicy.server.cxf.way.it.WssS
ecurityPolicyHelloServiceImpl
```

In this example, where your Camel route normally starts with **from("cxf:bean:myCxfHelloEndpoint")**, it now starts with **from("direct:myDirectEndpoint")**.

CEQ-5286 MTOM Attachments: Unmarshalling Error when invoking a CXF service with an AWT Image parameter

If your Service Endpoint Interface (SEI) refers to **java.awt.Image** in any of its method signatures, then the endpoint throws an error similar to the following:

- **{{Unmarshalling Error: unexpected element (uri:"", local:"arg0"). Expected elements are (none) }}**
- **Unmarshalling Error: unexpected element (uri:"https://quarkiverse.github.io/quarkiverse-docs/quarkus-cxf/test/mtom-awt", local:"data"). Expected elements are (none)**

You can workaround this issue by wrapping **java.awt.Image** inside a request or response data object. For example, you can configure your SEI as follows:

```
@WebService(name = "ImageService", targetNamespace = ImageService.NS)
@MTOM
public interface ImageService {
    public static final String NS = "https://quarkiverse.github.io/quarkiverse-docs/quarkus-
cxf/test/mtom-awt";

    @WebMethod
    Image downloadImage(@WebParam(name = "name", targetNamespace = NS) String name);

    @WebMethod
    String uploadImage(String name, Image image);
}
```

To use this workaround, you should configure your SEI as follows:

```
@WebService(name = "ImageService", targetNamespace = ImageService.NS)
@MTOM
public interface ImageService {
    public static final String NS = "https://quarkiverse.github.io/quarkiverse-docs/quarkus-
cxf/test/mtom-awt";

    @WebMethod
    ImageData downloadImage(@WebParam(name = "name", targetNamespace = NS) String name);

    @WebMethod
    String uploadImage(ImageData image);
}
```

```
@XmlType(name = "imageData", namespace = "http://org.jboss.ws/xop/doclit")
public class ImageData {
    private Image data;
    private String name;

    public ImageData() {
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public Image getData() {
        return data;
    }

    public void setData(Image data) {
        this.data = data;
    }
}
```

CEQ-4769 CXF clients that contain a class with postponed initialization in their method signatures cannot be compiled to native

When a client of Service Endpoint Interface (SEI) refers to **java.awt.Image** in any of its methods, the application cannot compile to native. There is no known workaround for this issue, but you can still use the affected clients in JVM mode.

2.5. IMPORTANT NOTES

Support for AdoptiumJDK

Camel Extensions for Quarkus version 2.13.2 introduces support for AdoptiumJDK 11 and AdoptiumJDK 17.

Camel upgraded from version 3.14.2 to version 3.18.3

Camel Extensions for Quarkus version 2.13.2 has been upgraded from Camel version 3.14.2 to Camel version 3.18.3. For additional information about each intervening Camel patch release, refer to the following:

- Apache Camel 3.14.3 Release Notes
- Apache Camel 3.14.4 Release Notes
- Apache Camel 3.14.5 Release Notes
- Apache Camel 3.14.6 Release Notes
- Apache Camel 3.14.7 Release Notes
- Apache Camel 3.15.0 Release Notes
- Apache Camel 3.16.0 Release Notes

- Apache Camel 3.17.0 Release Notes
- Apache Camel 3.18.0 Release Notes
- Apache Camel 3.18.1 Release Notes
- Apache Camel 3.18.2 Release Notes
- Apache Camel 3.18.3 Release Notes

Camel Quarkus upgraded from version 2.7 to version 2.13

Camel Extensions for Quarkus version 2.13.2 has been upgraded from Camel Quarkus version 2.7 to Camel Quarkus version 2.13. For additional information about each intervening Camel Quarkus patch release, refer to the following:

- Apache Camel Quarkus 2.8.0 Release Notes
- Apache Camel Quarkus 2.9.0 Release Notes
- Apache Camel Quarkus 2.10.0 Release Notes
- Apache Camel Quarkus 2.11.0 Release Notes
- Apache Camel Quarkus 2.12.0 Release Notes
- Apache Camel Quarkus 2.13.0 Release Notes
- Apache Camel Quarkus 2.13.1 Release Notes
- Apache Camel Quarkus 2.13.2 Release Notes

2.6. RESOLVED ISSUES

The following table lists known issues that were affecting Camel Extensions for Quarkus, which have been fixed in Camel Extensions for Quarkus version 2.13.2.

Table 2.1. Resolved issues

Issue	Description
CEQ-4769	Camel MLLP: Allow the ability to set MIN_BUFFER_SIZE for SocketBuffer.
CEQ-4754	Camel REST: Request URL duplicated context path.
CEQ-833	JMS components connection pooling.
CEQ-799	Request to support Camel Quarkus JTA+JPA integration.

For more details of other issues resolved between Camel Quarkus 2.7 and Camel Quarkus 2.13, see the Release Notes for each patch release.

2.7. EXTENSIONS ADDED IN THIS RELEASE

The following table lists the extensions that have been added in this release of Camel Extensions for Quarkus version 2.13.2.

Table 2.2. Added extensions

Extension	Artifact	Description
Browse	**camel-quarkus-browse**	Inspect the messages received on endpoints supporting BrowsableEndpoint.
CXF	**camel-quarkus-cxf-soap**	Expose SOAP WebServices using Apache CXF or connect to external WebServices using CXF WS client.
Dataformat	**camel-quarkus-dataformat**	Use a Camel Data Format as a regular Camel Component.
Google BigQuery	**camel-quarkus-google-bigquery**	Access Google Cloud BigQuery service using SQL queries or Google Client Services API.
Google Pubsub	**camel-quarkus-google-pubsub**	Send and receive messages to/from Google Cloud Platform PubSub Service.
Kubernetes	**camel-quarkus-kubernetes**	Perform operations against Kubernetes API.
REST OpenApi	**camel-quarkus-rest-openapi**	Configure REST producers based on an OpenAPI specification document delegating to a component implementing the RestProducerFactory interface.

2.8. DATA FORMATS ADDED IN THIS RELEASE

The following table lists the data formats that have been added in this release of Camel Extensions for Quarkus version 2.13.2.

Table 2.3. Added data formats

Extension	Artifact	Description
JAXB	**camel-quarkus-jaxb**	Unmarshal XML payloads to POJOs and back using JAXB2 XML marshalling standard.

2.9. ADDITIONAL RESOURCES

- Supported Configurations
- Camel Extensions for Quarkus
- Getting Started with Camel Extensions for Quarkus
- Developing Applications with Camel Extensions for Quarkus

CHAPTER 3. DEBEZIUM 2.1.3 RELEASE NOTES

Debezium is a distributed change data capture platform that captures row-level changes that occur in database tables and then passes corresponding change event records to Apache Kafka topics. Applications can read these *change event streams* and access the change events in the order in which they occurred. Debezium is built on Apache Kafka and is deployed and integrated with AMQ Streams on OpenShift Container Platform or on Red Hat Enterprise Linux.

The following topics provide release details:

- Section 3.1, “Debezium database connectors”
- Section 3.2, “Debezium supported configurations”
- Section 3.3, “Debezium installation options”
- Section 3.4, “Upgrading Debezium from version 1.x to 2.1.3”
- Section 3.5, “New Debezium features”
- Section 3.7, “Deprecated Debezium features”

3.1. DEBEZIUM DATABASE CONNECTORS

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

- Db2
- MongoDB
- MySQL
- Oracle
- PostgreSQL
- SQL Server

3.1.1. Connector usage notes

- Db2
    - The Debezium Db2 connector does not include the Db2 JDBC driver (**jcc-11.5.0.0.jar**). See the deployment instructions for information about how to deploy the necessary JDBC driver.
    - The Db2 connector requires the use of the abstract syntax notation (ASN) libraries, which are available as a standard part of Db2 for Linux.
    - To use the ASN libraries, you must have a license for IBM InfoSphere Data Replication (IIDR). You do not have to install IIDR to use the libraries.
- MongoDB
    - Currently, you cannot use the transaction metadata feature of the Debezium MongoDB connector with MongoDB 4.2.

- Oracle
  - The Debezium Oracle connector does not include the Oracle JDBC driver (**ojdbc8.jar**). See the deployment instructions for information about how to deploy the necessary JDBC driver.
- PostgreSQL
  - To use the Debezium PostgreSQL connector you must use the **pgoutput** logical decoding output plug-in, which is the default for PostgreSQL versions 10 and later.

Additional resources

- Getting Started with Debezium
- Debezium User Guide

3.2. DEBEZIUM SUPPORTED CONFIGURATIONS

For information about Debezium supported configurations, including information about supported database versions, see the Debezium 2.1.3 Supported configurations page .

3.2.1. AMQ Streams API version

Debezium runs on AMQ Streams 2.3.

AMQ Streams supports the **v1beta2** API version, which updates the schemas of the AMQ Streams custom resources. Older API versions are deprecated. After you upgrade to AMQ Streams 1.7, but before you upgrade to AMQ Streams 1.8 or later, you must upgrade your custom resources to use API version **v1beta2**.

For more information, see the Debezium User Guide.

3.3. DEBEZIUM INSTALLATION OPTIONS

You can install Debezium with AMQ Streams on OpenShift or on Red Hat Enterprise Linux:

- Installing Debezium on OpenShift
- Installing Debezium on RHEL

3.4. UPGRADING DEBEZIUM FROM VERSION 1.X TO 2.1.3

The current version of Debezium includes changes that require you to follow specific steps when you upgrade from an earlier version. For more information, refer to the list of breaking changes and the upgrade procedure.

3.4.1. Upgrading connectors to Debezium 2.1.3

Debezium 2.1.3 is the first Red Hat release of a new Debezium major release version. Some of the changes in the Debezium 2.1.3 are not backward-compatible with previous versions of Debezium. As a result, to preserve data and ensure continued operation when you upgrade from Debezium 1.x versions to 2.1.3, you must complete some manual steps during the upgrade process.

One significant change is that the names of some connector parameters have changed. To accommodate these changes, review the configuration properties updates, and note the properties that are present in your connector configuration. Before you upgrade, edit the configuration of each Debezium connector to add the new names of any changed properties. Before you upgrade, edit the configuration of any 1.x connector instances so that both the old and new property names are present. After the upgrade, you can remove the old configuration options.

Prerequisites

- Debezium is now compatible with Kafka versions up to 3.3.1. This is the default Kafka version in AMQ Streams 2.3.
- The Java 11 runtime is required and must be available prior to upgrading. AMQ Streams 2.3 supports Java 11. Use Java 11 when developing new applications. Java 11 enables use of recent language updates, such as the new String API and changes in predicate support, while also benefiting from Java performance improvements. Java 8 is no longer supported in AMQ Streams 2.3.
- Check the backward-incompatible changes in the Breaking changes list.
- Verify that your environment complies with the Debezium 2.1.3 Supported Configurations.

Procedure

1. From the OpenShift console, review the Kafka Connector YAML to identify the connector configuration that are no longer valid in Debezium 2.1.3. Refer to Table 3.1, “Updates to connector configuration properties” for details.
2. Edit the configuration to add the 2.x equivalents for the properties that you identify in Step 1, so that both the old and new property names are present. Set the values of the new properties to the values that were previously specified for the old properties.
3. From the OpenShift console, stop Kafka Connect to gracefully stop the connector.
4. From the OpenShift console, edit the Kafka Connect image YAML to reference the Debezium 2.1.3.Final version of the connector zip file.
5. From the OpenShift console, edit the Kafka Connector YAML to remove any configuration options that are no longer valid for your connector.
6. Adjust your application’s storage dependencies, as needed, depending on the storage module implementation dependencies in your code. See Changes to Debezium storage in the list of Breaking changes.
7. Restart Kafka Connect to start the connector. After you restart the connector, the 2.1.3.Final connector continues to process events from the point at which you stopped the connector before the upgrade. Change events records that the connector wrote to Kafka before the upgrade are not modified.

3.5. NEW DEBEZIUM FEATURES

Debezium 2.1.3 includes the following updates.

- Section 3.5.1, “Breaking changes”
- Section 3.5.2, “Features promoted to General Availability”

3.5.1. Breaking changes

The following changes in Debezium 2.1.3 represent significant differences in connector behavior and require configuration changes that are not compatible with earlier Debezium versions:

Changes that apply to multiple connectors

Database history topic

Now referred to as the *database schema history* topic.

Limits on object sizes for memory queues

Sizes are no longer calculated by using reflection. Instead, queue limits are estimated based on the message schema. (DBZ-2766) (MongodB, MySQL, Oracle, PostgreSQL, SQL Server)

Exposure of connector metrics

Debezium previously exposed connector metrics as a single tuple of snapshot, streaming, and history-based beans. With this release, connector metrics are now exposed as a multi-partition scheme. As a result, metrics names, and the way in which they are exposed is changed (DBZ-4726). If you use Grafana, Prometheus, or similar JMX frameworks for gathering metrics, review your process for collecting metrics.

database.server.name property

No longer used in the connector configuration. For more information, see Table 3.1, "Updates to connector configuration properties".

Schema definition

For naming and versioning consistency, Debezium schemas are now defined in a central point (DBZ-4365, DBZ-5044). If you use a schema registry, schema compatibility issues might occur.

Debezium storage changes

In previous releases, Integration supported reading and storing offsets, history, and other data as a part of the debezium-core module. This release includes a new **debezium-storage** module with implementations for storing data in a local file system or in Kafka (DBZ-5229). The extension point implemented in this approach makes it possible to introduce other storage implementations in the future. As part of the upgrade, you might need to adjust your application's dependencies depending on the storage module implementations required by the code.

Restart after communication exceptions

After an exception related to communication (SqlException, IOException) is thrown, by default, the Debezium MongoDB, MySQL, PostgreSQL, and SQL Server connectors now restart automatically (DBZ-5244).

Default value of the**skipped.operations** configuration option

The default value is now **truncate** (DBZ-5497) (MongoDB, MySQL, Oracle, PostgreSQL, SQL Server)

Default value of**schema.name.adjustment.mode** property

The default value is now **none** (DBZ-5541). The previous default option, **avro** was a good choice for customers who use the Avro converter, but it caused confusion in environments that use the JSON converter. As part of this change, the **sanitize.field.names** property is no longer available.

* Removal of connector configuration properties

Several properties that were available in Debezium 1.x versions are no longer valid and have been replaced by new properties. For more information, see the following table:

Table 3.1. Updates to connector configuration properties

1.x property	Equivalent 2.x property
database.* (pass-through database driver properties) (DBZ-5043)	**driver.***
database.dbname (SQL Server)	**database.names**
database.history.consumer.* (DBZ-5043)	**schema.history.internal.consumer.***
database.history.kafka.bootstrap.servers (DBZ-5043)	**schema.history.internal.kafka.bootstrap.servers**
database.history.kafka.topic (DBZ-5043)	**schema.history.internal.kafka.topic**
database.history.producer.* (DBZ-5043)	**schema.history.internal.producer.***
database.server.name (DBZ-5043)	**topic.prefix**
mongodb.name (MongoDB)	**topic.prefix**
schema_blacklist (DBZ-5045)	**schema_exclude_list**
schema_whitelist (DBZ-5045)	**schema_include_list**

Changes that apply to the MySQL connector

- The MySQL connector no longer supports legacy JDBC legacy date/time properties (DBZ-4965).

Changes that apply to the MongoDB connector

- The MongoDB connector no longer supports streaming directly from the oplog. Change streams represents a superior mechanism for performing change data capture with MongoDB. Rather than reading the oplog directly, the connector now delegates the task of capturing and decoding the oplog data to MongoDB change streams, which expose the changes that occur within a collection as an event stream. The Debezium connector subscribes to the stream and then delivers the changes downstream to Kafka. The transition to change streams offers a variety of benefits, including the ability to stream changes from non-primary nodes, and the ability to emit update events with a full document representation for downstream consumers.

- The configuration property **mongodb.name** is replaced by the **topic.prefix** property.

Changes that apply to the PostgreSQL connector

- Protocol buffer (**protobuf**) decoding is no longer supported (DBZ-703).

- The **wal2json** plugin is no longer supported (DBZ-4156).

- The PostgreSQL transaction id is now 32-bit integer that rolls over. To simplify de-duplication of transactions, the LSN is now included as part of the identifier (DBZ-5329).

Changes that apply to the SQL Server connector

- If SSL is not enabled for a SQL Server database, or if you want to connect to the database without using SSL, disable SSL by setting the value of the **database.encrypt** property in the connector configuration to **false**.
- The **database.dbname** property is replaced by the **database.names** property.

3.5.2. Features promoted to General Availability

The following features are promoted from Technology Preview to General Availability in the Debezium 2.1.3 release:

3.5.3. Debezium feature updates

This Debezium 2.1.3 release provides several feature updates and fixes, including the items in the following list:

3.6. TECHNOLOGY PREVIEW FEATURES

IMPORTANT

Technology Preview features are not supported with Red Hat production service-level agreements (SLAs) and might not be functionally complete. Red Hat does not recommend implementing any Technology Preview features in production environments. Technology Preview features provide 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.

Debezium includes the following Technology Preview features:

Parallel initial snapshots

You can optionally configure SQL-based connectors to use multiple threads when performing an initial snapshot by setting the **snapshot.max.threads** property to a value greater than 1.

Ad hoc and incremental snapshots for MongoDB connector

Provides a mechanism for re-running a snapshot of a table for which you previously captured a snapshot.

CloudEvents converter

Emits change event records that conform to the CloudEvents specification. The CloudEvents change event envelope can be JSON or Avro and each envelope type supports JSON or Avro as the **data** format. The CloudEvents change event envelope supports Avro encoding change event envelope can be JSON or Avro and each envelope type supports JSON or Avro as the **data** format.

Content-based routing

Provides a mechanism for rerouting selected events to specific topics, based on the event content.

Custom-developed converters

In cases where the default data type conversions do not meet your needs, you can create custom converters to use with a connector.

Filter SMT

Enables you to specify a subset of records that you want the connector to send to the broker.

Signaling for the MongoDB connector

Provides a mechanism for modifying the behavior of a connector, or triggering a one-time action, such as initiating an ad hoc snapshot of a table.

Use of the BLOB, CLOB, and NCLOB data types with the Oracle connector

The Oracle connector can consume Oracle large object types.

3.7. DEPRECATED DEBEZIUM FEATURES

PostgreSQL truncate.handling.mode property

The **truncate.handling.mode** property for the Debezium PostgreSQL connector is deprecated in this release and is scheduled for removal in a future release (DBZ-4419). Use the **skipped.operations** property in its place.

MonitoredTables option for connector snapshot and streaming metrics

The **MonitoredTables** option for Debezium connector metrics is deprecated in this release and scheduled for removal in a future release. Use the **CapturedTables** metric in its place.

CHAPTER 4. CAMEL K RELEASE NOTES

Camel K is a lightweight integration framework built from Apache Camel K that runs natively in the cloud on OpenShift. Camel K is specifically designed for serverless and microservice architectures. You can use Camel K to instantly run integration code written in Camel Domain Specific Language (DSL) directly on OpenShift.

Using Camel K with OpenShift Serverless and Knative, containers are automatically created only as needed and are autoscaled under load up and down to zero. This removes the overhead of server provisioning and maintenance and enables you to focus instead on application development.

Using Camel K with OpenShift Serverless and Knative Eventing, you can manage how components in your system communicate in an event-driven architecture for serverless applications. This provides flexibility and creates efficiencies using a publish/subscribe or event-streaming model with decoupled relationships between event producers and consumers.

4.1. CAMEL K FEATURES

The Camel K provides cloud-native integration with the following main features:

- Knative Serving for autoscaling and scale-to-zero
- Knative Eventing for event-driven architectures
- Performance optimizations using Quarkus runtime by default
- Camel integrations written in Java or YAML DSL
- Monitoring of integrations using Prometheus in OpenShift
- Quickstart tutorials
- Kamelet Catalog for connectors to external systems such as AWS, Jira, and Salesforce
- Support for Timer and Log Kamelets
- Metering for Camel K Operator and pods
- Support for IBM MQ connector
- Support for Oracle 19 database

4.2. SUPPORTED CONFIGURATIONS

For information about Camel K supported configurations, standards, and components, see the following Customer Portal articles:

- Camel K Supported Configurations
- Camel K Component Details

4.2.1. Camel K Operator metadata

The Camel K includes updated Operator metadata used to install Camel K from the OpenShift OperatorHub. This Operator metadata includes the Operator bundle format for release packaging, which is designed for use with OpenShift Container Platform 4.6 or later.

Additional resources

- Operator bundle format in the OpenShift documentation .

4.3. IMPORTANT NOTES

Important notes for the Red Hat Integration - Camel K release:

Support to run Camel K on ROSA

Camel K is now supported to run on Red Hat OpenShift Service on AWS (ROSA).

Support for IBM MQ source connector in Camel K

IBM MQ source connector kamelet is added to latest Camel K.

Support for Oracle 19

Oracle 19 is now supported in Camel K. Refer Supported configurations page for more information.

Using Camel K CLI commands on Windows machine

When using kamel cli commands on Windows machine, the path in the **resource** option in the command must use linux format. For example,

```
//Windows path
kamel run file.groovy --dev --resource file:C:\user\folder\tempfile@/tmp/file.txt

//Must be converted to
kamel run file.groovy --dev --resource file:C:/user/folder/tempfile@/tmp/file.txt
```

Red Hat Integration - Camel K Operator image size is increased

Since Red Hat Integration - Camel K 1.10.0.redhat-00033, the size of the Camel K Operator image is doubled.

Accepted Camel case notations in YAML DSL

Since Red Hat Integration - Camel K 1.10.0.redhat-00033, the YAML DSL will accept camel case notation (i.e, **setBody**) as well as snake case (i.e **set-body**). Please note that there are some differences in the syntax as schema is subject to changes within Camel versions.

4.4. SUPPORTED CAMEL QUARKUS EXTENSIONS

This section lists the Camel Quarkus extensions that are supported for this release of Camel K (only when used inside a Camel K application).

NOTE

These Camel Quarkus extensions are supported only when used inside a Camel K application. These Camel Quarkus extensions are not supported for use in standalone mode (without Camel K).

4.4.1. Supported Camel Quarkus connector extensions

The following table shows the Camel Quarkus connector extensions that are supported for this release of Camel K (only when used inside a Camel K application).

Name	Package
AWS 2 Kinesis	**camel-quarkus-aws2-kinesis**
AWS 2 Lambda	**camel-quarkus-aws2-lambda**
AWS 2 S3 Storage Service	**camel-quarkus-aws2-s3**
AWS 2 Simple Notification System (SNS)	**camel-quarkus-aws2-sns**
AWS 2 Simple Queue Service (SQS)	**camel-quarkus-aws2-sqs**
Azure Storage Blob (Technology Preview)	**camel-quarkus-azure-storage-blob**
Azure Storage Queue (Technology Preview)	**camel-quarkus-azure-storage-queue**
Cassandra CQL	**camel-quarkus-cassandraql**
File	**camel-quarkus-file**
FTP	**camel-quarkus-ftp**
FTPS	**camel-quarkus-ftp**
SFTP	**camel-quarkus-ftp**
HTTP	**camel-quarkus-http**
JMS	**camel-quarkus-jms**
Kafka	**camel-quarkus-kafka**
Kamelets	**camel-quarkus-kamelet**
Metrics	**camel-quarkus-microprofile-metrics**
MongoDB	**camel-quarkus-mongodb**
Salesforce	**camel-quarkus-salesforce**
SQL	**camel-quarkus-sql**
Timer	**camel-quarkus-timer**

4.4.2. Supported Camel Quarkus dataformat extensions

The following table shows the Camel Quarkus dataformat extensions that are supported for this release of Camel K (only when used inside a Camel K application).

Name	Package
Avro	**camel-quarkus-avro**
Bindy (for CSV)	**camel-qaurkus-bindy**
Gson	**camel-quarkus-gson**
JSON Jackson	**camel-quarkus-jackson**
Jackson Avro	**camel-quarkus-jackson-avro**

4.4.3. Supported Camel Quarkus language extensions

In this release, Camel K supports the following Camel Quarkus language extensions (for use in Camel expressions and predicates):

- Constant
- ExchangeProperty
- File
- Header
- Ref
- Simple
- Tokenize
- JsonPath

4.4.4. Supported Camel K traits

In this release, Camel K supports the following Camel K traits.

- Builder trait
- Camel trait
- Container trait
- Dependencies trait
- Deployer trait
- Deployment trait

- Environment trait
- Jvm trait
- Kamelets trait
- Owner trait
- Platform trait
- Pull Secret trait
- Prometheus trait
- Quarkus trait
- Route trait
- Service trait
- Error Handler trait

4.5. SUPPORTED KAMELETS

The following table lists the kamelets that are provided as OpenShift resources when you install the Camel K operator.

For details about these kamelets, go to: https://github.com/openshift-integration/kamelet-catalog/tree/kamelet-catalog-1.8

For information about how to use kamelets to connect applications and services, see https://access.redhat.com/documentation/en-us/red_hat_integration/2022.q3/html-single/integrating_applications_with_kamelets.

IMPORTANT

Technology Preview features are not supported with Red Hat production service level agreements (SLAs) and might not be functionally complete. Red Hat does not recommend using them in production.

These features provide early access to upcoming product features, enabling customers to test functionality and provide feedback during the development process. For more information about the support scope of Red Hat Technology Preview features, see https://access.redhat.com/support/offerings/techpreview.

Table 4.1. Kamelets provided with the Camel K operator

Kamelet	File name	Type (Sink, Source, Action)
Ceph sink	**ceph-sink.kamelet.yaml**	Sink
Ceph Source	**ceph-source.kamelet.yaml**	Source

Kamelet	File name	Type (Sink, Source, Action)
Jira Add Comment sink	**jira-add-comment-sink.kamelet.yaml**	Sink
Jira Add Issue sink	**jira-add-issue-sink.kamelet.yaml**	Sink
Jira Transition Issue sink	**jira-transition-issue-sink.kamelet.yaml**	Sink
Jira Update Issue sink	**jira-update-issue-sink.kamelet.yaml**	Sink
Avro Deserialize action	**avro-deserialize-action.kamelet.yaml**	Action (data conversion)
Avro Serialize action	**avro-serialize-action.kamelet.yaml**	Action (data conversion)
AWS DynamoDB sink	**aws-ddb-sink.kamelet.yaml**	Sink
AWS Redshift sink	**aws-redshift-sink.kamelet.yaml**	Sink
AWS 2 Kinesis sink	**aws-kinesis-sink.kamelet.yaml**	Sink
AWS 2 Kinesis source	**aws-kinesis-source.kamelet.yaml**	Source
AWS 2 Lambda sink	**aws-lambda-sink.kamelet.yaml**	Sink
AWS 2 Simple Notification System sink	**aws-sns-sink.kamelet.yaml**	Sink
AWS 2 Simple Queue Service sink	**aws-sqs-sink.kamelet.yaml**	Sink
AWS 2 Simple Queue Service source	**aws-sqs-source.kamelet.yaml**	Source
AWS 2 Simple Queue Service FIFO sink	**aws-sqs-fifo-sink.kamelet.yaml**	Sink
AWS 2 S3 sink	**aws-s3-sink.kamelet.yaml**	Sink
AWS 2 S3 source	**aws-s3-source.kamelet.yaml**	Source
AWS 2 S3 Streaming Upload sink	**aws-s3-streaming-upload-sink.kamelet.yaml**	Sink
Azure Storage Blob Source (Technology Preview)	**azure-storage-blob-source.kamelet.yaml**	Source

Kamelet	File name	Type (Sink, Source, Action)
Azure Storage Blob Sink (Technology Preview)	**azure-storage-blob-sink.kamelet.yaml**	Sink
Azure Storage Queue Source (Technology Preview)	**azure-storage-queue-source.kamelet.yaml**	Source
Azure Storage Queue Sink (Technology Preview)	**azure-storage-queue-sink.kamelet.yaml**	Sink
Cassandra sink	**cassandra-sink.kamelet.yaml**	Sink
Cassandra source	**cassandra-source.kamelet.yaml**	Source
Extract Field action	**extract-field-action.kamelet.yaml**	Action
FTP sink	**ftp-sink.kamelet.yaml**	Sink
FTP source	**ftp-source.kamelet.yaml**	Source
Has Header Key Filter action	**has-header-filter-action.kamelet.yaml**	Action (data transformation)
Hoist Field action	**hoist-field-action.kamelet.yaml**	Action
HTTP sink	**http-sink.kamelet.yaml**	Sink
Insert Field action	**insert-field-action.kamelet.yaml**	Action (data transformation)
Insert Header action	**insert-header-action.kamelet.yaml**	Action (data transformation)
Is Tombstone Filter action	**is-tombstone-filter-action.kamelet.yaml**	Action (data transformation)
Jira source	**jira-source.kamelet.yaml**	Source
JMS sink	**jms-amqp-10-sink.kamelet.yaml**	Sink
JMS source	**jms-amqp-10-source.kamelet.yaml**	Source

Kamelet	File name	Type (Sink, Source, Action)
JMS IBM MQ sink	**jms-ibm-mq-sink.kamelet.yaml**	Sink
JMS IBM MQ source	**jms-ibm-mq-source.kamelet.yaml**	Source
JSON Deserialize action	**json-deserialize-action.kamelet.yaml**	Action (data conversion)
JSON Serialize action	**json-serialize-action.kamelet.yaml**	Action (data conversion)
Kafka sink	**kafka-sink.kamelet.yaml**	Sink
Kafka source	**kafka-source.kamelet.yaml**	Source
Kafka Topic Name Filter action	**topic-name-matches-filter-action.kamelet.yaml**	Action (data transformation)
Log sink (for development and testing purposes)	**log-sink.kamelet.yaml**	Sink
MariaDB sink	**mariadb-sink.kamelet.yaml**	Sink
Mask Fields action	**mask-field-action.kamelet.yaml**	Action (data transformation)
Message TimeStamp Router action	**message-timestamp-router-action.kamelet.yaml**	Action (router)
MongoDB sink	**mongodb-sink.kamelet.yaml**	Sink
MongoDB source	**mongodb-source.kamelet.yaml**	Source
MySQL sink	**mysql-sink.kamelet.yaml**	Sink
PostgreSQL sink	**postgresql-sink.kamelet.yaml**	Sink
Predicate filter action	**predicate-filter-action.kamelet.yaml**	Action (router/filter)
Protobuf Deserialize action	**protobuf-deserialize-action.kamelet.yaml**	Action (data conversion)

Kamelet	File name	Type (Sink, Source, Action)
Protobuf Serialize action	**protobuf-serialize-action.kamelet.yaml**	Action (data conversion)
Regex Router action	**regex-router-action.kamelet.yaml**	Action (router)
Replace Field action	**replace-field-action.kamelet.yaml**	Action
Salesforce Create	**salesforce-create-sink.kamelet.yaml**	Sink
Salesforce Delete	**salesforce-delete-sink.kamelet.yaml**	Sink
Salesforce Update	**salesforce-update-sink.kamelet.yaml**	Sink
SFTP sink	**sftp-sink.kamelet.yaml**	Sink
SFTP source	**sftp-source.kamelet.yaml**	Source
Slack source	**slack-source.kamelet.yaml**	Source
SQL Server Database sink	**sqlserver-sink.kamelet.yaml**	Sink
Telegram source	**telegram-source.kamelet.yaml**	Source
Throttle action	**throttle-action.kamelet.yaml**	Action
Timer source (for development and testing purposes)	**timer-source.kamelet.yaml**	Source
TimeStamp Router action	**timestamp-router-action.kamelet.yaml**	Action (router)
Value to Key action	**value-to-key-action.kamelet.yaml**	Action (data transformation)

4.6. CAMEL K KNOWN ISSUES

The following known issues apply to the Camel K:

ENTESB-15306 - CRD conflicts between Camel K and Fuse Online

If an older version of Camel K has ever been installed in the same OpenShift cluster, installing Camel K from the OperatorHub fails due to conflicts with custom resource definitions. For example, this includes older versions of Camel K previously available in Fuse Online.

For a workaround, you can install Camel K in a different OpenShift cluster, or enter the following command before installing Camel K:

```
$ oc get crds -l app=camel-k -o json | oc delete -f -
```

ENTESB-15858 - Added ability to package and run Camel integrations locally or as container images

Packaging and running Camel integrations locally or as container images is not currently included in the Camel K and has community-only support.

For more details, see the Apache Camel K community .

ENTESB-16477 - Unable to download jira client dependency with productized build

When using Camel K operator, the integration is unable to find dependencies for jira client. The work around is to add the atlassian repo manually.

```
apiVersion: camel.apache.org/v1
kind: IntegrationPlatform
metadata:
  labels:
    app: camel-k
  name: camel-k
spec:
  configuration:
  - type: repository
    value: <atlassian repo here>
```

ENTESB-17033 - Camel-K ElasticsearchComponent options ignored

When configuring the Elasticsearch component, the Camel K ElasticsearchComponent options are ignored. The work around is to add **getContext().setAutowiredEnabled(false)** when using the Elasticsearch component.

ENTESB-17061 - Can’t run mongo-db-source kamelet route with non-admin user - Failed to start route mongodb-source-1 because of null

It is not possible to run **mongo-db-source kamelet** route with non-admin user credentials. Some part of the component require admin credentials hence it is not possible run the route as a non-admin user.

4.7. CAMEL K FIXED ISSUES

The following sections list the issues that have been fixed in Red Hat Integration - Camel K 1.10.0.redhat-00033.

- Section 4.7.1, “Feature requests in Camel K 1.10.0.redhat-00033”
- Section 4.7.2, “Enhancements in Camel K 1.10.0.redhat-00033”
- Section 4.7.3, “Bugs resolved in Camel K 1.10.0.redhat-00033”

4.7.1. Feature requests in Camel K 1.10.0.redhat-00033

The following table lists the feature requests in Camel K 1.10.0.redhat-00033.

Table 4.2. Camel K 1.10.0.redhat-00033 feature requests

Issue	Description
ENTESB-17612	Add repeatCount property to timer-source kamelet
ENTESB-17753	Performance testing Camel K for inclusion with Developer Sandbox
ENTESB-19480	Camel-K: Support for Camel master component
ENTESB-19718	Sync with upstream
ENTESB-20372	consumerGroup property support by kafka-source Kamelet

4.7.2. Enhancements in Camel K 1.10.0.redhat-00033

The following table lists the enhancements in Camel K 1.10.0.redhat-00033.

Table 4.3. Camel K 1.10.0.redhat-00033 Enhancements

Issue	Description
ENTESB-16272	Provide full Custom Resource Definition Structural schema for Camel K Traits
ENTESB-17659	Expose parameter for mongo-db camel component to specify user auth DB
ENTESB-18950	Support scale sub-resource in Dev Sandbox idler
ENTESB-19636	Add support for jslt-action kamelet
ENTESB-19720	Onboard the cpaas productization of camel-k to OSBS 2.0
ENTESB-19965	Azure Storage Blob Sink and Source Kamelet: Adding credentialsType parameter to both

4.7.3. Bugs resolved in Camel K 1.10.0.redhat-00033

The following table lists the resolved bugs in Camel K 1.10.0.redhat-00033.

Table 4.4. Camel K 1.10.0.redhat-00033 Resolved Bugs

Issue	Description
ENTESB-16801	Camel-K 1.4.0: camel-jackson Unrecognized Type: [null]
ENTESB-17050	CVE-2021-30129 sshd-sftp: mina-sshd-core: Memory leak denial of service in Apache Mina SSHD Server [rhint-camel-k-1]

Issue	Description
ENTESB-17582	CVE-2021-37136 netty-codec: Bzip2Decoder doesn’t allow setting size restrictions for decompressed data [rhint-camel-k-1]
ENTESB-17584	CVE-2021-37137 netty-codec: SnappyFrameDecoder doesn’t restrict chunk length and may buffer skippable chunks in an unnecessary way [rhint-camel-k-1]
ENTESB-17601	dataformat dependency not resolved
ENTESB-17759	Knative broker to AWS SQS Sink Number of message attributes exceeds the allowed maximum
ENTESB-17843	Kameletbinding from broker source ignores broker name
ENTESB-18014	CK MRRC zip - missing manifests for jars
ENTESB-18021	Missing source jars in CK MRRC
ENTESB-18288	CVE-2021-42550 logback-classic: logback: remote code execution through JNDI call from within its configuration file [rhint-camel-k-1]
ENTESB-18295	Source zip files and m2 zip files present in CK MRRC
ENTESB-18299	target directory present in the CK source zip
ENTESB-18487	CVE-2021-22569 protobuf-java: potential DoS in the parsing procedure for binary data [rhint-camel-k-1]
ENTESB-18535	CVE-2021-41571 pulsar-client-admin: pulsar: Pulsar Admin API allows access to data from other tenants using getMessageById API [rhint-camel-k-1]
ENTESB-18579	CVE-2022-23596 junrar: A carefully crafted RAR archive can trigger an infinite loop while extracting [rhint-camel-k-1]
ENTESB-18582	CVE-2021-43859 xstream: Injecting highly recursive collections or maps can cause a Denial of Service [rhint-camel-k-1]
ENTESB-18588	CVE-2022-21724 quarkus-jdbc-postgresql-deployment: jdbc-postgresql: Unchecked Class Instantiation when providing Plugin Classes [rhint-camel-k-1]
ENTESB-18683	CVE-2022-23913 artemis-commons: Apache ActiveMQ Artemis Denial of Service [rhint-camel-k-1]
ENTESB-18688	CVE-2022-0981 quarkus: privilege escalation vulnerability with RestEasy Reactive scope leakage in Quarkus [rhint-camel-k-1]

Issue	Description
ENTESB-19005	Camel K operator putting memory pressure on Kube ApiServer
ENTESB-19339	CVE-2022-2053 undertow: Large AJP request may cause Denial of Service [rhint-camel-k-1]
ENTESB-19394	Camel-K 1.8: Integration not marked as failed when it cannot be built.
ENTESB-19485	Windows CLI: option --resource do not work with windows path system (\ as separator)
ENTESB-19490	CVE-2022-33980 commons-configuration2: apache-commons-configuration: Apache Commons Configuration insecure interpolation defaults [rhint-camel-k-1]
ENTESB-19671	The supported trait jolokia doesn't use productized artifacts
ENTESB-19713	CVE-2022-25857 snakeyaml: Denial of Service due to missing nested depth limitation for collections [rhint-camel-k-1]
ENTESB-19960	1.8.1 is using an older openjdk image than 1.6.10
ENTESB-19963	CVE-2022-40154 xstream: Xstream to serialize XML data was vulnerable to Denial of Service attacks [rhint-camel-k-1]
ENTESB-19969	CVE-2022-40156 xstream: Xstream to serialize XML data was vulnerable to a Denial of Service attacks [rhint-camel-k-1]
ENTESB-19971	CVE-2022-40155 xstream: Xstream to serialize XML data was vulnerable to a Denial of Service attacks [rhint-camel-k-1]
ENTESB-19973	CVE-2022-40153 xstream: Xstream to serialize XML data was vulnerable to a Denial of Service attacks [rhint-camel-k-1]
ENTESB-19975	CVE-2022-40152 woodstox-core: woodstox to serialize XML data was vulnerable to Denial of Service attacks [rhint-camel-k-1]
ENTESB-19977	CVE-2022-40151 xstream: Xstream to serialize XML data was vulnerable to Denial of Service attacks [rhint-camel-k-1]
ENTESB-19992	CVE-2022-38751 snakeyaml: Uncaught exception in java.base/java.util.regex.Pattern$Ques.match [rhint-camel-k-1]
ENTESB-19994	CVE-2022-38750 snakeyaml: Uncaught exception in org.yaml.snakeyaml.constructor.BaseConstructor.constructObject [rhint-camel-k-1]

Issue	Description
ENTESB-19996	CVE-2022-38749 snakeyaml: Uncaught exception in org.yaml.snakeyaml.composer.Composer.composeSequenceNode [rhint-camel-k-1]
ENTESB-19999	CVE-2022-42889 commons-text: apache-commons-text: variable interpolation RCE [rhint-camel-k-1]
ENTESB-20002	CVE-2022-42003 jackson-databind: deep wrapper array nesting wrt UNWRAP_SINGLE_VALUE_ARRAYS [rhint-camel-k-1]
ENTESB-20004	CVE-2022-42004 jackson-databind: use of deeply nested arrays [rhint-camel-k-1]
ENTESB-20098	Camel-cloudevents missing from build
ENTESB-20403	Ready condition message not always taken from Camel Health Check
ENTESB-20486	Camel K: gc trait fails for: cannot list resource \"endpoints\" in API group \"submariner.io\"
ENTESB-20488	CVE-2022-40149 jettison: parser crash by stackoverflow [rhint-camel-k-1]
ENTESB-20499	KameletBinding with native annotation fails to build a quarkus app with "Out of date version of GraalVM detected"
ENTESB-20501	Wrong default OLM channel with kamel install
ENTESB-20502	no matches for kind \"PodDisruptionBudget\" in version \"policy/v1beta1\""
ENTESB-20603	[serialize/deserialize-action kamelet] Failed to start application: Unsupported field: property-name
ENTESB-20606	CVE-2022-37866 apache-ivy: Apache Ivy: Ivy Path traversal [rhint-camel-k-1]
ENTESB-20634	CVE-2022-38648 batik: Server-Side Request Forgery [rhint-camel-k-1]
ENTESB-20635	CVE-2022-38398 batik: Server-Side Request Forgery [rhint-camel-k-1]
ENTESB-20636	CVE-2022-40146 batik: Server-Side Request Forgery (SSRF) vulnerability [rhint-camel-k-1]
ENTESB-20649	CVE-2022-45693 jettison: If the value in map is the map itself, the new JSONObject(map) can cause StackOverflowError, which may lead to a Denial of Service. [rhint-camel-k-1]

Issue	Description
ENTESB-20657	Regression from 1.8 - Kameletbinding with dataformat action kamelets produces warning CDI: programmatic lookup problem detected
ENTESB-20659	Regression from 1.8 - body type changed to StreamCache from byte[] for field action kamelets
ENTESB-20660	Kamelet native build: Fatal error: com.oracle.graal.pointsto.util.AnalysisError$ParsingError
ENTESB-20706	[jackson-databind] Vulnerable artifact present in MRRC

CHAPTER 5. CAMEL SPRING BOOT 3.14.5 PATCH 01 RELEASE NOTES

5.1. CAMEL SPRING BOOT FEATURES

Camel Spring Boot introduces Camel support for Spring Boot which provides auto-configuration of the Camel and starters for many Camel components. The opinionated auto-configuration of the Camel context auto-detects Camel routes available in the Spring context and registers the key Camel utilities (like producer template, consumer template and the type converter) as beans.

5.2. SUPPORTED PLATFORMS, CONFIGURATIONS, DATABASES, AND EXTENSIONS FOR CAMEL SPRING BOOT

- For information about supported platforms, configurations, and databases in Camel Spring Boot, see the Supported Configuration page on the Customer Portal (login required).
- For a list of Red Hat Camel Spring Boot extensions, see the *Camel Spring Boot Reference* (login required).

5.3. IMPORTANT NOTES

Documentation for Camel Spring Boot components is available in the Camel Spring Boot Reference. Documentation for additional Camel Spring Boot components will be added to this reference guide.

Migration from Fuse 7.11 to Camel Spring Boot

This release contains a Migration Guide documenting the changes required to successfully run and deploy Fuse 7.11 applications on Camel Spring Boot. It provides information on how to resolve deployment and runtime problems and prevent changes in application behavior. Migration is the first step in moving to the Camel Spring Boot platform. Once the application deploys successfully and runs, users can plan to upgrade individual components to use the new functions and features of Camel Spring Boot.

Support for EIP circuit breaker

The Circuit Breaker EIP for Camel Spring Boot supports Resilience4j configuration. This configuration provides integration with Resilience4j to be used as Circuit Breaker in Camel routes.

5.4. CAMEL SPRING BOOT FIXED ISSUES

The following sections list the issues that have been fixed in Camel Spring Boot.

- Section 5.4.1, “Camel Spring Boot version 3.14.5 Patch 01 Fixed Issues”

5.4.1. Camel Spring Boot version 3.14.5 Patch 01 Fixed Issues

The following table lists the resolved bugs in Camel Spring Boot version 3.14.5 Patch 01.

Table 5.1. Camel Spring Boot version 3.14.5 Patch 01 Resolved Bugs

Issue	Description
CSB-757	Missing cxf artifacts from MRRC

Issue	Description
CSB-788	Lack of artifacts(org.apache.cxf.cxf-spring-boot-starter-jaxws.3.4.8.redhat-00034) in maven.repository.redhat.com repository
CSB-793	CVE-2022-40149 jettison: parser crash by stackoverflow [rhint-camel-spring-boot-3]
CSB-927	CVE-2022-46364 CXF: Apache CXF: SSRF Vulnerability [rhint-camel-spring-boot-3]
CSB-928	CVE-2022-46363 CXF: Apache CXF: directory listing/code exfiltration [rhint-camel-spring-boot-3]
CSB-948	CVE-2022-45693 jettison: If the value in map is the map itself, the new JSONObject(map) can cause a StackOverflowError, which may lead to a Denial of Service [rhint-camel-spring-boot-3]

5.5. ADDITIONAL RESOURCES

- Supported Configurations
- Camel Spring Boot Reference
- Getting Started with Camel Spring Boot
- Migration Guide

CHAPTER 6. SERVICE REGISTRY RELEASE NOTES

Service Registry 2.3 is provided as a General Availability release. Service Registry is a datastore for standard event schemas and API designs, and is based on the Apicurio Registry open source community project.

You can use Service Registry to manage and share the structure of your data using a web console, REST API, Maven plug-in, or Java client. For example, client applications can dynamically push or pull the latest schema updates to or from Service Registry without needing to redeploy. You can also create optional rules to govern how Service Registry content evolves over time. These rules include content validation and backwards or forwards compatibility of schema or API versions.

6.1. SERVICE REGISTRY INSTALLATION OPTIONS

You can install Service Registry on OpenShift with either of the following data storage options:

- PostgreSQL database
- Red Hat AMQ Streams

For more details, see Installing and deploying Service Registry on OpenShift .

6.2. SERVICE REGISTRY SUPPORTED PLATFORMS

Service Registry 2.3 supports the following platform component versions:

- Red Hat OpenShift Container Platform 4.8 - 4.12
- Red Hat OpenShift Service on AWS
- Microsoft Azure Red Hat OpenShift
- PostgreSQL 12 - 15
- Red Hat AMQ Streams 2.1 - 2.3
- Red Hat Single Sign-On (RH-SSO) 7.6
- OpenJDK 11

6.3. SERVICE REGISTRY NEW FEATURES

Service Registry 2.3 includes the following new features:

Service Registry authentication and authorization

- *Expanded role-based authorization* - you can now configure role-based authorization in Service Registry, as well as in RH-SSO as previously. If role-based authorization is enabled in the Service Registry application, you can use the web console or REST API to control access.
- *Expanded owner-based authorization* - you can now enable the owner-based authorization option at the artifact-group level, as well as at the artifact level as previously.
- *Anonymous read access* - when the anonymous read access option is enabled, unauthenticated (anonymous) users have read-only access to all artifacts.

- *Authenticated read access* - when the authenticated read access option is enabled, any authenticated user has read-only access to all artifacts, even if the user has not been granted any Service Registry roles.

- *HTTP basic authentication* - when this option is enabled, users or client applications can use HTTP basic authentication to access Service Registry.

- *Custom TLS certificate for Kafka storage* - when using Kafka for storage, users can now securely connect to Kafka using a custom TLS certificate.

- *Change artifact owner* - administrators or artifact owners can change the owner of a specific schema or API artifact by using the REST API or web console.

Operational and monitoring improvements

- *Audit logging* - any changes to Service Registry data result in an audit log entry.

- *Prometheus metrics* - metrics are exposed in Prometheus format for use in monitoring.

- *Sentry integration* - optional integration with Sentry 1.x.

Operator improvements

- *Custom environment variables* - you can now set arbitrary environment variables in the **ApicurioRegistry** custom resource. These variables are applied to Service Registry using the **Deployment** resource.

- *Support for PodDisruptionBudget* - This resource is automatically created to ensure that at most one replica is unavailable.

- *Support for NetworkPolicy* - the Service Registry Operator creates an ingress network policy for port 8080.

Artifact references

Artifacts can now reference other artifacts in Service Registry. Many supported artifact types allow references from one file to another. For example, an OpenAPI file might have a data type with a property that references a JSON schema defined in another file. Typically, these references have a syntax specific to the artifact type. You can now use the REST API to create mappings so that type-specific references can be resolved to artifacts registered in Service Registry.

Dynamic global configuration of Service Registry instances

Service Registry has many global configuration options that are typically set at deployment time. A subset of these options are now also configurable at runtime for a Service Registry instance. You can manage these options at runtime by using the REST API or web console. For example, these options include owner-based authorization, anonymous read access, and authenticated read access.

Upload artifact from URL

You can now upload a schema or API artifact from a URL, in addition to the already supported upload from a file. You can upload by using the Service Registry web console or the REST API.

Web console improvements

- *Import and export of Service Registry data* - admin users can now use the web console to export all Service Registry data in a **.zip** file, as well as using the REST API as previously. They can then import this **.zip** file into a different Service Registry deployment.

- *Full support for artifact properties* - artifacts in Service Registry can have user-defined and editable metadata such as name, description, labels (simple keyword list), and properties (name/value pairs). The web console has been enhanced to support displaying and editing properties, in addition to using the REST API as previously.

- *Documentation generation for AsyncAPI artifacts* - AsyncAPI artifacts now support the **Documentation** tab on the artifact details page. This tab displays human-readable documentation generated from the AsyncAPI content. This feature was previously available only for OpenAPI artifacts.

- *Option to display JSON as YAML* - for artifact types that are JSON formatted, the **Content** tab on the artifact details page now supports switching between JSON and YAML formats.

REST API improvements

- *Improved /users/me endpoint* - the Service Registry core REST API has a **/users/me** endpoint that returns information about the current authenticated user. You can use this endpoint to inspect a user's assigned role and determine their capabilities.

- *Updated support for Confluent Compatibility API* - Service Registry now supports the Confluent Schema Registry API version 6.

Service Registry user documentation and examples

The documentation library has been updated with the new features available in version 2.3:

- Installing and deploying Service Registry on OpenShift
- Migrating Service Registry deployments
- Service Registry User Guide
- Apicurio Registry v2 core REST API documentation

The open source demonstration applications have also been updated:

- https://github.com/Apicurio/apicurio-registry-examples

6.4. SERVICE REGISTRY DEPRECATED FEATURES

Service Registry version 1.x

Service Registry version 1.x was deprecated in version 2.0 and is no longer fully supported. For more details, see the Red Hat Application Services Product Update and Support Policy .

6.5. UPGRADING AND MIGRATING SERVICE REGISTRY DEPLOYMENTS

You can upgrade automatically from Service Registry 2.0 to Service Registry 2.3 on OpenShift. There is no automatic upgrade from Service Registry 1.x to Service Registry 2.x, and a migration process is required.

6.5.1. Upgrading a Service Registry 2.0 deployment on OpenShift

You can upgrade from Service Registry 2.0.3 on OpenShift 4.9 to Service Registry 2.3.x on OpenShift 4.11 or later. You must upgrade both your Service Registry and your OpenShift versions, and upgrade OpenShift one minor version at a time.

Prerequisites

- You already have Service Registry 2.0.3 installed on OpenShift 4.9.

Procedure

1. In the OpenShift Container Platform web console, click **Administration** and then **Cluster Settings**.
2. Click the pencil icon next to the **Channel** field, and select the next minor **candidate** version (for example, change from **stable-4.9** to **candidate-4.10**).
3. Click **Save** and then **Update**, and wait until the upgrade is complete.
4. If the OpenShift version is less than 4.11, repeat steps 2 and 3, and select **candidate-4.11** or later.
5. Click **Operators** > **Installed Operators** > **Red Hat Integration - Service Registry**.
6. Ensure that the **Update channel** is set to **2.x**.
7. If the **Update approval** is set to **Automatic**, the upgrade should be approved and installed immediately after the **2.x** channel is set.
8. If the **Update approval** is set to **Manual**, click **Install**.
9. Wait until the Operator is deployed and the Service Registry pod is deployed.
10. Verify that your Service Registry system is up and running.

Additional resources

- For more details on how to set the Operator update channel in the OpenShift Container Platform web console, see Changing the update channel for an Operator.

6.5.2. Migrating a Service Registry 1.1 deployment on OpenShift

For details on migrating a Service Registry 1.1 deployment to Service Registry 2.x, see Migrating Service Registry deployments.

6.6. SERVICE REGISTRY RESOLVED ISSUES

Table 6.1. Service Registry resolved issues in version 2.3.0

Issue	Description
Registry-2394	REST API endpoint for core v1 compatibility not properly protected by authentication.
Registry-1959	Web console incorrectly redirects to HTTP instead of HTTPS.
Registry-1926	Service Registry throws **io.apicurio.registry.storage.ArtifactNotFoundException** while uploading a new artifact.

Issue	Description
Registry-1905	Confluent compatibility layer not working with JSON Schema artifacts.
Registry-1873	**kafkasql** registry storage option throws **Expected one element, but found none** exception when querying **contentIdFromHash**.
Registry-1660	Confluent compatibility layer's schema DTO is not fully compatible.
Registry-1610	Web console does not properly obey the disable roles feature.
Registry-1593	Confluent compatibility API v6 does not return artifact.
Registry- 733	Passing **sasl.jaas.config** property does not work with**JAVA_OPTIONS** environment variable.
Registry-651	Web console displays inconsistent **modifiedOn** date.
Registry-358	Global compatibility rule execution broken for **RuleApplicationType.CREATE**.
Registry-342	Transitive compatibility rules might give false positives.

Table 6.2. Service Registry resolved issues in version 2.3.3

Issue	Description
IPT-858	Avro compatibility check does not work correctly for **enum** types.
Registry-3128	Add option to minify Avro with Service Registry Maven plug-in.
Registry-3121	Make max subjects configurable in Confluent compatibility API.
Registry-3080	Throw exception when an empty schema is provided in the Confluent compatibility API.
Registry-3014	Fix handling of default JSON value in the Confluent compatibility API.
Registry-2991	On slow machines, **kafkasql** storage is not ready for existing messages.
Registry-2952	Fix version ordering in Service Registry compatibility rules.
Registry-2919	Support **application/json** in registry export API operation.
Registry-2913	Configuring Service Registry event sourcing gives HTTP error.
Registry-2877	Protobuf schema version upload failing with **NullPointerException**.

6.7. SERVICE REGISTRY RESOLVED CVES

Table 6.3. Service Registry resolved Common Vulnerabilities and Exposures (CVEs) in version 2.3.x

Issue	Description
IPT-789	CVE-2022-25858 terser: Insecure use of regular expressions leads to ReDoS.
IPT-788	CVE-2022-37734 graphql-java: DoS by malicious query.
IPT-787	CVE-2022-25857 snakeyaml: DoS due to missing nested depth limitation for collections.
IPT-764, IPT-763	CVE-2022-31129 moment: Inefficient parsing algorithm resulting in DoS.
IPT-760	CVE-2022-25647 com.google.code.gson-gson: Deserialization of untrusted data.
IPT-739	CVE-2022-24773 node-forge: Signature verification leniency in checking **DigestInfo** structure.
IPT-738	CVE-2022-24772 node-forge: Signature verification failing to check tailing garbage bytes can lead to signature forgery.
IPT-737	CVE-2022-24771 node-forge: Signature verification leniency in checking **digestAlgorithm** structure can lead to signature forgery.
IPT-734	CVE-2022-26520 jdbc-postgresql: Arbitrary file write vulnerability.
IPT-733	CVE-2022-0536 follow-redirects: Exposure of sensitive information by **Authorization Header** leak.
IPT-732	CVE-2022-0235 node-fetch: Exposure of sensitive information to an unauthorized actor.
IPT-731	CVE-2022-23647 prismjs: Improperly escaped output allows an XSS vulnerability.
IPT-728	CVE-2022-0981 quarkus: Privilege escalation vulnerability with RestEasy Reactive scope leakage in Quarkus.
IPT-723	CVE-2022-21724 quarkus-jdbc-postgresql-deployment: Unchecked class instantiation when providing plug-in classes.
IPT-705	CVE-2021-22569 protobuf-java: Potential DoS in the parsing procedure for binary data.
IPT-652	CVE-2021-41269 cron-utils: Template injection leading to unauthenticated Remote Code Execution vulnerability.

Issue	Description
IPT-566	CVE-2021-37136 netty-codec: **Bzip2Decoder** doesn't allow setting size restrictions for decompressed data.
IPT-564	CVE-2021-37137 netty-codec: **SnappyFrameDecoder** doesn't restrict chunk length and might buffer skippable chunks in an unnecessary way.

6.8. SERVICE REGISTRY KNOWN ISSUES

The following known issues apply in Service Registry 2.3.3:

Service Registry core known issues

IPT-814 - Service Registry logout feature incompatible with RH-SSO 7.6

In RH-SSO 7.6, the **redirect_uri** parameter used with the logout endpoint is deprecated. For more details, see the RH-SSO 7.6 Upgrading Guide . Because of this deprecation, when Service Registry is secured by using the RH-SSO Operator, clicking the **Logout** button displays the **Invalid parameter: redirect_uri** error.

For a workaround, see https://access.redhat.com/solutions/6980926.

IPT-701 - CVE-2022-23221 H2 allows loading custom classes from remote servers through JNDI

When Service Registry data is stored in AMQ Streams, the H2 database console allows remote attackers to execute arbitrary code by using the JDBC URL. Service Registry is not vulnerable by default and a malicious configuration change is required.

Service Registry Operator known issues

Operator-42 - Autogeneration of OpenShift route might use wrong base host value

If multiple **routerCanonicalHostname** values are specified, autogeneration of the Service Registry OpenShift route might use a wrong base host value.