Red Hat Integration 2023.Q4

Red Hat Integration 2023.q4 发行注记

Red Hat Integration 中的新功能

Last Updated: 2024-04-11

Red Hat Integration 2023.Q4 Red Hat Integration 2023.q4 发行注记

Red Hat Integration 中的新功能

法律通告

摘要

描述 Red Hat Integration 产品，并提供了有关本版本中新内容的最新详情。

目录

前言

使开源包含更多

红帽致力于替换我们的代码、文档和 Web 属性中存在问题的语言。我们从这四个术语开始：master、slave、黑名单和白名单。由于此项工作十分艰巨，这些更改将在即将推出的几个发行版本中逐步实施。详情请查看 CTO Chris Wright 的信息。

第 1 章 RED HAT INTEGRATION

Red Hat Integration 是一组全面的集成和事件处理技术，用于在混合和多环境中创建、扩展和部署基于容器的集成服务。Red Hat Integration 提供了一个敏捷、分布式和以 API 为中心的解决方案，组织可用于连接和共享数字世界中所需的应用程序和系统之间的数据。

Red Hat Integration 包括以下功能：

- 实时消息传递
- 跨数据中心消息流
- API 连接
- 应用程序连接器
- 企业级集成模式
- API 管理
- 数据转换
- 服务组成和编配

其他资源

- 了解企业集成

第 2 章 DEBEZIUM 2.3.4 发行注记

Debezium 是一个分布式更改数据捕获平台，用于捕获数据库表中发生的行级更改，然后将对应的更改事件记录传递给 Apache Kafka 主题。应用程序可以读取 *这些更改事件流*，并按发生更改事件的顺序访问更改事件。Debezium 基于 Apache Kafka 构建，并在 OpenShift Container Platform 或 Red Hat Enterprise Linux 上部署并与 AMQ Streams 集成。

以下主题提供发行版本详情：

2.1. DEBEZIUM 数据库连接器

Debezium 为以下通用数据库提供基于 Kafka Connect 的连接器：

- Db2
- JDBC Sink 连接器（开发者预览）
- MongoDB
- MySQL
- Oracle
- PostgreSQL
- SQL Server

2.1.1. 连接器使用备注

- Db2
  - Debezium Db2 连接器不包含 Db2 JDBC 驱动程序(**jcc-11.5.0.0.jar**)。有关如何部署必要 JDBC 驱动程序的信息，请参阅 Db2 连接器部署说明。
  - Db2 连接器需要使用抽象语法表示法(ASN)库，这些库作为 Linux 的 Db2 标准部分提供。
  - 要使用 ASN 库，您必须有用于 IBM InfoSphere 数据复制(IIDR)的许可证。您不必安装 IIDR 来使用库。
- Oracle
  - Debezium Oracle 连接器不包含 Oracle JDBC 驱动程序(**ojdbc8.jar**)。有关如何部署必要 JDBC 驱动程序的详情，请查看 Oracle 连接器部署说明。

- PostgreSQL
  - 要使用 Debezium PostgreSQL 连接器，您必须使用 **pgoutput** 逻辑解码输出插件，这是 PostgreSQL 版本 10 及更高版本的默认设置。

其他资源

- Debezium 入门
- Debezium 用户指南

2.2. DEBEZIUM 支持的配置

有关 Debezium 支持的配置的详情，包括支持的数据库版本的信息，请参阅 Debezium 2.3.4 支持的配置页面。

2.2.1. AMQ Streams API 版本

Debezium 在 AMQ Streams 2.5 上运行。

AMQ Streams 支持 **v1beta2** API 版本，它更新 AMQ Streams 自定义资源的 schema。旧的 API 版本已弃用。升级到 AMQ Streams 1.7 后，但在升级到 AMQ Streams 1.8 或更高版本前，您必须升级自定义资源以使用 API 版本 **v1beta2**。

如需更多信息，请参阅 Debezium 用户指南。

2.3. DEBEZIUM 安装选项

您可以在 OpenShift 或 Red Hat Enterprise Linux 中使用 AMQ Streams 安装 Debezium：

- 在 OpenShift 上安装 Debezium
- 在 RHEL 上安装 Debezium

2.4. 将 DEBEZIUM 从版本 1.X 升级到 2.3.4

当前版本的 Debezium 包括在从 2.1.4 之前的版本升级时，需要您执行特定步骤的更改。

2.4.1. 将连接器升级到 Debezium 2.3.4

Debezium 2.3.4 发行版本包括一些与早于 2.x 的 Debezium 版本不兼容的一些更改。因此，在从 Debezium 1.x 版本升级到 2.3.4 时保留数据并确保继续操作，您必须在升级过程中完成一些手动步骤。

一个显著的更改是某些连接器参数的名称已更改。要容纳这些更改，请查看 2023.Q2 发行注记中的 配置属性更新，并记录您的连接器配置中存在的属性。在升级前，编辑每个 Debezium 连接器的配置，以添加任何更改的属性的新名称。在升级前，编辑任何 1.x 连接器实例的配置，以便存在旧属性和新属性名称。升级后，您可以删除旧的配置选项。

前提条件

- Debezium 现在与 Kafka 版本兼容，最多 3.5.0。这是 AMQ Streams 2.5 中的默认 Kafka 版本。

- 需要 Java 11 运行时，必须在升级前使用。AMQ Streams 2.5 支持 Java 11。在开发新应用程序时使用 Java 11。Java 11 允许使用最新的语言更新，如新的 String API 和 predicate 支持的变化，同时受益于 Java 性能改进。AMQ Streams 2.5 不再支持 Java 8。
- 检查当前 更改以及 2023.Q2 发行注记中的与向后兼容性相关的更改。
- 验证您的环境是否符合 Debezium 2.3.4 支持的配置。

流程

1. 在 OpenShift 控制台中，查看 Kafka Connector YAML 来识别在 Debezium 2.3.4 中不再有效的连接器配置。详情请参阅 2023.Q2 发行注记。
2. 编辑配置，为在第 1 步中识别的属性添加 2.x 等效项，以便存在旧属性和新属性名称。将新属性的值设置为之前为旧属性指定的值。
3. 在 OpenShift 控制台中，停止 Kafka Connect 以安全停止连接器。
4. 在 OpenShift 控制台中，编辑 Kafka Connect 镜像 YAML 来引用连接器 zip 文件的 Debezium 2.3.4.Final 版本。
5. 在 OpenShift 控制台中，编辑 Kafka Connector YAML 以删除不再对连接器有效的配置选项。
6. 根据需要调整应用程序的存储依赖项，具体取决于代码中的存储模块实现依赖项。如需更多信息，请参阅 2023.Q2 发行注记中的 Debezium 存储的变化。
7. 重启 Kafka Connect 以启动连接器。重启连接器后，它会从升级前停止的时间点继续处理事件。在不修改升级前，更改连接器写入 Kafka 的事件记录。

2.5. 新功能及改进

Debezium 2.3.4 包括以下更新和改进：

- 可能会造成问题的更改
- 提升到正式发行(GA)的功能
- 正式发行(GA)功能
- 技术预览功能
- 开发人员预览功能
- 本发行版本中的其他更新

2.5.1. 可能会造成问题的更改

Debezium 2.3.4 中的以下变化代表连接器行为的显著区别，需要与早期 Debezium 版本不兼容的配置更改： Debezium 2.3.4 引入了以下破坏更改：

- MySQL 和 PostgreSQL 安全连接更改
- 主题和模式命名更改
- Oracle 连接器的源 info 块更改

有关破坏上一个 Debezium 发行版本中更改的详情，请参考 2023.Q2 发行注记。

2.5.1.1. MySQL 和 PostgreSQL 安全连接的新配置默认值

您可以为 MySQL 和 PostgreSQL 配置 Debezium 连接器，以使用安全 SSL 连接。对于 MySQL 连接器，您可以通过配置 **database.ssl.mode** 属性来指定使用安全连接。对于 PostgreSQL 连接器，您可以设置 **database.sslmode** 属性。

从 Debezium 2.3.4 开始，这些配置选项包括新的默认值。对于 MySQL，现在 **首选 database.ssl.mode** 的默认值，替换之前的默认值 **disabled**。对于 PostgreSQL，现在 **首选 database.sslmode** 的默认值，替换之前的默认值 **disable**。根据新的默认设置，当连接器启动与数据库的连接时，它们首先会尝试建立加密的安全连接。如果没有可用的安全连接，连接器会返回使用不安全的连接，除非另有配置。

2.5.1.2. 主题和模式命名更改

当 Debezium 生成主题名称和模式名称时，它会替换名称中的非 ASCII 字符，以确保与 schema registry 的命名约定兼容。在早期版本中，Debezium 替换了下划线字符(_)来替换非 ASCII 字符。然而，在某些情况下，在替换非 ASCII 字符后，Debezium 为两个主题或模式生成的名称可能相同，但其字母校准除外，这可能会导致其他问题。

为了以最兼容的方式解决这个问题，Debezium 现在使用基于策略的方法来映射唯一字符。这个新方法的一个副作用是 Debezium 不再支持 **sanitize.field.names** 配置属性。在 **sanitize.field.names** 属性的位置，现在可使用新选项来指定与用于表或集合的约定兼容的命名策略。

要指定 Debezium 如何生成 schema 和字段名称，您可以设置以下属性。

schema.name.adjustment.mode

指定应如何调整模式名称以与消息转换器兼容。

field.name.adjustment.mode

指定应如何调整字段名称以与消息转换器兼容。

对于前面的每个属性，您可以设置以下值之一：

none

名称按原样传递；不会对模式或字段名称进行调整。

avro

将 Avro 中使用的字符替换为下划线(_)。

avro_unicode

将 Avro 中使用的下划线(_)和字符替换为基于 unicode 的转义序列。

2.5.1.3. 对 Oracle 连接器源信息块的更改

Debezium 发送用于插入、更新和删除事件的更改事件记录包含一个包含 **源** 信息块的有效负载。对于 Oracle 连接器，源信息块包含一个代表更改 SQL 序列号的特殊 sn 字段。

在某些情况下，源数据库中 ssn 字段的值超过 INT32 数据类型(2,147,483,647)的最大值。为了允许较大的值，Debezium 现在将数据类型 INT64 分配给 sn 字段，这会将字段的最大值增加到 9,223,372,036,854,775,807。

如果您当前将 sn 值存储在您的环境中的 sink 系统中，或者您使用 schema registry，则这个更改可能会影响系统的行为。

2.5.2. 提升到正式发行(GA)的功能

在 Debezium 2.3.4 发行版本中，以下功能从技术预览提升到正式发行(GA)：

MongoDB 连接器的临时和增量快照

提供用于重新运行之前捕获快照的表的快照的机制。

MongoDB 连接器的信号 https://access.redhat.com/documentation/zh-cn/red_hat_integration/2023.q4/html-single/debezium_user_guide/index#sending-signals-to-a-debezium-connector

提供修改连接器行为或触发一次性操作的机制，如启动表 的临时快照。

基于内容的路由

提供根据事件内容将所选事件重新路由到特定主题的机制。

过滤 SMT

允许您指定希望连接器发送到代理的记录子集。

2.5.3. 正式发行(GA)功能

Debezium 2.3.4 支持以下新功能：

- PostgreSQL 的自动副本身份配置
- 新通知子系统（接收器、日志、JMX）
- 关联增量快照通知 ID
- 支持新的信号频道（源、Kafka、JMX）

- Oracle RAC 的改进
- Oracle 连接器基于 SCN 的指标
- MongoDB 和 Oracle 的服务器端过滤
- 启动时重试数据库连接
- 增量快照的 Srogate 密钥
- ExtractChangedRecordState SMT
- ExtractNewRecordState (event flattening) SMT 选项，用于从事件记录中删除字段
- HeaderToValue SMT
- 分区路由 SMT

2.5.3.1. PostgreSQL 的自动副本身份配置

Debezium 2.3.4 引入了一个新的 PostgreSQL 连接器功能，称为 "Autoset Replica Identity"。

PostgreSQL 数据库使用副本身份来识别在数据库事务日志中捕获的列，以进行插入、更新和删除事件。此功能允许您将连接器配置为自动为表设置副本身份值。当连接器启动时，它会读取副本身份配置，然后为指定表设置副本身份。

新的配置属性 replica.identity.autoset.values 指定以逗号分隔的表和副本身份元组列表。当属性指定表的副本身份时，该值会覆盖任何现有的副本身份配置。有关 PostgreSQL 副本身份类型的更多信息，请参阅 PostgreSQL 文档。

replica.identity.autoset.values 属性接受以逗号分隔的值列表，每个元素使用 <fully-qualified-table-name>:<replica-identity> 格式。以下示例演示了如何配置两个表(table1 和 table2)来具有 FULL

副本身份：

```
{ "replica.identity.autoset.values": "public.table1:FULL,public.table2:FULL" }
```

连接器访问数据库的用户帐户需要权限来设置表的副本身份。如果帐户缺少足够权限，则任何尝试使用 replica.identity.autoset.values 会导致失败。如果无法使用属性自动设置副本身份，则必须从具有所需权限的数据库帐户手动设置表的副本身份。

2.5.3.2. 新的通知子系统

此发行版本引入了一个新的通知子系统，它允许 Debezium 发送报告各种连接器操作状态的事件，如增量或传统快照。这个新子系统允许您通过几个不同的频道发送通知，包括 Kafka 主题、日志文件和 Java 管理扩展(JMX)。这些通知事件可由各种外部系统使用。通知事件以一系列键/值元组表示，包括以下字段：

id

用于标识通知的 UUID，

aggregate_type

根据域驱动的设计的概念，通知类型。

type

提供有关聚合类型的更多详细信息。

additional_data （可选）

基于字符串的键/值对映射，其中包含有关事件的额外信息。

以下示例显示了一个简单的通知事件。

通知事件示例

```
{
  "id": "c485ccc3-16ff-47cc-b4e8-b56a57c3bad2",
  "aggregate_type": "Snapshot",
  "type": "Started",
  "additional_data": {
```

```
    ...
  }
}
```

在本发行版本中，Debebe 支持以下通知事件类型：

- 初始快照的状态
- 增量快照进度

如需更多信息，请参阅配置通知以报告连接器状态。

2.5.3.3. 关联增量快照通知 ID

在本发行版本中，通知和频道子系统已被改进，将信号与通知相关联。也就是说，当您发送信号并且由 Debezium 使用时，生成的通知包含一个引用原始信号的标识符。当通信在多个应用程序和进程之间分布时，这种机制使进程能够更轻松地将信号与其生成的操作相关联。

2.5.3.4. 支持新的信号频道

Debezium 具有支持的信号，因为版本 1.7 中引入了增量快照。信号提供了一种使用元数据的机制来指示 Debezium 执行任务，如将条目写入连接器日志或执行临时增量快照。

此发行版本引进了对多个信号频道的支持，允许您指定 Debezium 用来监视并响应信号的介质。在以前的版本中，一个频道在连接器之间被通用支持，这是数据库信号表。在这个发行版本中，默认提供以下内容：

- 数据库信号表
- Kafka 信号主题
- JMX

在本发行版本中，信号频道子系统已被改进，以支持通过 JMX 发送信号。在 JConsole 窗口中，连接器、通知部分和 signal 部分现在存在两个子部分。

signal 部分允许您对 JMX bean 调用操作，来向 Debezium 传输信号。这个信号类似于它接受 3 参数的逻辑信号表结构：

- 唯一标识符
- 信号类型
- 信号有效负载。

如需更多信息，请参阅 向集成连接器发送信号

2.5.3.5. Oracle RAC 的改进

当您将 Debezium Oracle 连接器与 Oracle Real Application Clusters (RAC)部署搭配使用时，您必须指定一个 rac.nodes 配置属性。至少，rac.nodes 属性必须指定集群中每个节点的主机或 IP 地址。连接器的旧版本也支持备用格式，您可以在其中为每个节点指定唯一的端口号，以标识不同节点可能使用不同的端口的事实。

Debezium 2.3.4 通过识别每个节点可能使用不同的 Oracle 站点标识符(SID)来提高 Oracle RAC 支持。要考虑 Oracle SID 配置中的变体，您可以在 rac.nodes 配置属性中指定 SID 参数。

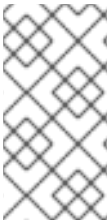
以下示例演示了连接到两个 Oracle RAC 节点，每个节点使用不同的端口和 SID 参数：

```
{ "connector.class": "io.debezium.connector.oracle.oracleConnector", "rac.nodes":
"host1.domain.com:1521/ORCLSID1,host2.domain.com:1522/ORCLSID2", ... }
```

2.5.3.6. Oracle 连接器基于 SCN 的指标

Oracle 在其 JMX 指标中跟踪各种系统更改号(SCN)值，包括 OffsetScn、currentScn、SOldestScn 和 CommittedScn。这些 SCN 值是数字，通常不能超过 LONG 数据类型的上限。在过去的版本中，Debezium 公开了 SCN 值作为 String 值。

为提高这些指标的实用程序，Debezium 现在将这些 JMX 指标公开为 BigInteger 值，而不是 String 值。这个更改可让用户通过不支持基于字符串的值的工具（如 Grafana 和 Prometheus）查看这些指标的值。



注意

如果您之前为其他目的收集 SCN 值，请注意它们不再基于字符串，且必须解释为 BigInteger 数字值。

2.5.3.7. MongoDB 和 Oracle 连接器的服务器端过滤

从数据库获取条目时，MongoDB 和 Oracle 连接器现在可以提交在连接器配置中设置的 include 和 exclude 过滤器。MongoDB 连接器会自动执行此操作。如果您希望 Oracle 连接器在获取条目时提交过滤器，请将 log.mining.query.filter.mode 属性设为 none 以外的值，这是默认值。

在过去，MongoDB 和 Oracle 连接器首先从数据库中获取事件，然后根据过滤器设置评估事件。这个过程有效地将数据库的所有更改序列化到连接器。一种效率低下的方法，特别是在高容量环境中。连接器因为过滤设置而只收到一些事件来立即丢弃它们。对于在云环境中运行的连接器，传输此类大量数据降低利用率成本。

要减少连接器获取的数据量，在 Debezium 2.3.4 中，连接器不会在获取数据后评估过滤器。相反，include 和 exclude 列表在 MongoDB 更改流订阅或 Oracle 获取查询中定义。通过减少连接器读取的事件数量，新的方法会降低网络和 CPU 使用率。对于使用完整文档或预镜像设置配置的连接器，这会为完全不需要的网络添加更多利用率。另外，通过启用连接器只接收需要处理的事件，连接器可以完成更多处理，从而增加 CPU 使用率。

2.5.3.8. 在连接器启动过程中重试数据库连接

在以前的版本中，连接器在启动时使用快速策略。也就是说，如果连接器无法执行完成启动例程所需的任何步骤，例如连接到数据库或验证，连接器将进入 FAILED 状态。

在某些情况下，连接器可能会正常启动，在一段时间内运行，然后最终遇到严重错误。错误可能与连接器启动生命周期中未访问的资源相关，以便在不遇到错误的情况下重启连接器。但是，当因为数据库因数据库不可用时，如果数据库重启后不可用，fail-fast 策略会导致连接器进入 FAILED 状态。然后，您必须手动干预来解决这个问题。

为提高可靠性和弹性，在此版本中，不会试图在启动时访问潜在的不可用资源，连接器现在会在其生命周期后尝试访问这些资源。实际上，在启动期间，Debezium 对访问潜在不可用资源的限制不太严格，使其能够利用 Kafka Connect 重试后端框架。

现在，如果数据库在连接器启动过程中不可用，只要启用了 Kafka Connect 重试，连接器将继续重试失败的请求。只有在达到最大重试尝试次数后，或者发生不可修改的错误时，才会生成 FAILED 状态。

2.5.3.9. 在增量快照中使用 Surrogate 键

Debezium 增量快照功能提供了一种执行可恢复、一致的数据快照的机制。对于必须最佳大量数据的连接器，恢复快照的能力可能至关重要。

在早期版本中，增量快照要求为快照中包含的每个表设置了主键。从 Debezium 2.3.4 开始，您现在可以在无密钥表上执行增量快照，只要表包含一个可以充当"surrogate 键"的唯一快照。

警告

对增量快照使用 Surrogate 键的功能只适用于 Debezium 关系连接器；您不能将此功能与 MongoDB 连接器一起使用。

要在增量快照信号中提供 surrogate 键列数据，您必须在信号有效负载中包含新的 surrogate key 属性和 surrogate-key。

指定 surrogate 键的增量快照信号有效负载示例

```
{
  "data-collections": [ "public.mytab" ],
  "surrogate-key": "customer_ref"
}
```

上例中的信号启动表 public.mytab 的增量快照。快照使用 customer_ref 列作为生成快照窗口的主键。

警告

您必须使用单个列来定义 surrogate 键。您无法定义基于多个列的 surrogate 键。

您还可以将 surrogate 键功能与具有主密钥的表一起使用。例如，当表的主键由多个列组成时，Surrogate 键具有优势。基于多个列的查询会为主键中的每个列生成一个 disjunction predicate，性能可能会高度依赖于环境。使用 surrogate 键减少查询中的列数可以提供更统一的性能。

使用 surrogate 键还可以为主键列基于基于字符的数据类型的表提供优势。由于关系数据库在进行数字比较与字符比较时通常更高效，因此您可以通过指定数字存活键来提高查询性能。

2.5.3.10. ExtractChangedRecordState SMT

此发行版本引入了事件记录更改(ExtractChangedRecordState)单一消息转换(SMT)。您可以使用此转换来识别 Debezium 事件记录中值更改或在数据库操作后保持不变的字段。要使用转换，请将其配置为连接器配置的一部分，例如：

```
transforms=changes
transforms.changes.type=io.debezium.transforms.ExtractChangedRecordState
transforms.changes.header.changed=ChangedFields
transforms.changes.header.unchanged=UnchangedFields
```

您可以为这个转换设置以下选项以指示不同类型的更改：

header.changed

显示事件更改的字段。

header.unchanged

显示事件保持不变的字段。

如上例所示，您可以将这两个选项设置为单独显示 changed 和 changed 字段。

转换会添加带有指定名称的新标头，如 ChangedFields。然后，它会将标头值设置为包含 changed

或 changed 字段名称的列表。

有关使用 ExtractChangedRecordState SMT 的更多信息，请参阅 Debezium 用户指南中的 事件记录更改。

2.5.3.11. 使用 ExtractNewRecordState SMT 的新配置选项丢弃事件字段

您可以使用 ExtractNewRecordState 单一消息转换(SMT)将 Debezium 更改事件转换为由 sink 连接器使用的简化格式。

此发行版本添加了三个用于转换的新配置选项，可用于丢弃事件有效负载或消息键中的字段：

drop.fields.header.name

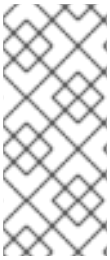
用于列出要丢弃的源消息中的字段名称的 Kafka 消息标头名称。

drop.fields.from.key

指定是否也从键中删除字段，默认为 false。

drop.fields.keep.schema.compatible

指定是否删除只是可选的的字段，默认为 true。



注意

为了保持使用 Avro 的环境中的模式兼容性，SMT 默认为强制模式兼容性。因此，如果您将必填字段配置为丢弃，则 SMT 不会从键或有效负载中删除该字段，除非您禁用模式兼容性。

发送仅包含更改的字段的事件

您可以使用更新的 ExtractNewRecordState SMT 对 ExtractChangedRecordState 转换进行对，将连接器配置为发送仅包含更改的字段的事件。以下示例显示了仅在事件的 payload 值中发出 changed 列的配置：

```
transforms=changes,extract
transforms.changes.type=io.debezium.transforms.ExtractChangedRecordState
transforms.changes.header.unchanged=UnchangedFields
transforms.extract.type=io.debezium.transforms.ExtractNewRecordState
transforms.extract.drop.fields.header.name=UnchangedFields
```

以上配置列出了未更改的字段，但 ir 不会将它们从事件有效负载中删除。如果指定键中的字段没有改变，则会保留它，因为配置不会明确更改 drop.fields.from.key 的默认 false 值。

如果 SMT 会导致在事件有效负载中丢弃必填字段，因为它没有更改，以遵守模式兼容性，则会在输出中保留该字段。

有关 ExtractNewRecordState SMT 的更多信息，请参阅 Debezium 更改事件后在状态 后 提取源记录。

2.5.3.12. HeaderToValue SMT

从事件记录中提取指定的标头字段，然后将标头字段复制或移到事件记录中的值。如需更多信息，请参阅 Debezium 用户指南中的 将消息标头转换为事件记录值。

2.5.3.13. 分区路由 SMT

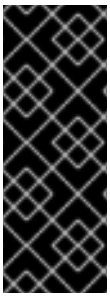
PartitionRouting SMT 可让您根据一个或多个指定有效负载字段的值将事件路由到特定的目标分区。要计算目标分区，Debebe 会生成指定字段值的哈希值。

如需更多信息，请参阅 Debezium 用户指南中的 基于 payload 字段将记录路由到分区。

2.5.4. 技术预览功能

此发行版本包括以下技术预览功能：

- 第 2.5.4.1 节 “MongoDB sharded 集群改进（技术预览）”
- 第 2.5.4.2 节 “用于多副本和分片集群的 MongoDB 增量快照（技术预览）”



重要

技术预览功能不受红帽产品服务等级协议（SLA）支持，且功能可能并不完整。红帽不推荐在生产环境中实施任何技术预览功能。技术预览功能为用户提供了一个对最新的产品创新的试用机会，以便用户可以对其进行测试并提供反馈。如需有关支持范围的更多信息，请参阅 技术预览功能支持范围。

2.5.4.1. MongoDB sharded 集群改进（技术预览）

在过去的版本中，当您在分片集群部署中使用 Debezium MongoDB 连接器时，连接器会打开与分片中设置的每个副本直接连接。这个方法与 MongoDB 冲突建议连接器应该 打开与 mongos 路由器实例的连接。

在这个发行版本中，连接器已被重新设计，以使用推荐的连接策略。如果您在分片集群中使用连接器，请调整您的配置，以便连接器连接到 mongos 实例。不需要其他更改。

2.5.4.2. 用于多副本和分片集群的 MongoDB 增量快照（技术预览）

现在，您可以将增量快照与 MongoDB 多副本集群一起使用。如需更多信息，请参阅 {NameUserGuide} 的 MongoDB 章节中的增加快照。

以前提供技术预览功能

之前版本中引入的以下功能仍为技术预览：

并行初始快照

您可选择将基于 SQL 的连接器配置为在执行初始快照时使用多个线程，方法是将 snapshot.max.threads 属性设置为大于 1 的值。

CloudEvents converter

发出更改事件，记录符合 CloudEvents 规格。CloudEvents 更改事件 envelope 可以是 JSON 或 Avro，每个 envelope 类型都支持 JSON 或 Avro 作为 数据格式。

自定义开发的转换器

如果默认数据类型转换无法满足您的需要，您可以创建自定义转换器以用于连接器。

使用带有 Oracle 连接器的 BLOB、CLOB 和 NCLOB 数据类型

Oracle 连接器可以使用 Oracle 大对象类型。

2.5.5. 开发人员预览功能

此 Debezium 2.3.4 发行版本包括以下开发人员技术预览功能：

- 第 2.5.5.1 节 “JDBC Sink Connector （开发者预览）”
- 第 2.5.5.2 节 “MySQL 连接器：并行快照（开发者预览）”
- 第 2.5.5.4 节 “PostgreSQL 连接器的一次交付（开发者预览）”
- 第 2.5.5.3 节 “Oracle 连接器：从 Oracle 逻辑待机中获得更改（开发者预览）”

重要

红帽以任何方式支持开发人员预览功能，且功能不完整或生产就绪。对于生产环境或关键业务工作负载，不要使用开发人员预览软件。开发人员预览软件可提前访问即将发布的产品软件。客户可以使用此软件测试功能并在开发过程中提供反馈。此软件可能没有任何文档，可以随时更改或删除，并收到有限的测试。红帽可能会提供在没有关联的 SLA 的情况下提交开发人员预览软件反馈的方法。

有关 Red Hat Developer Preview 软件支持范围的更多信息，请参阅 开发人员预览支持范围。

2.5.5.1. JDBC Sink Connector （开发者预览）

在此发行版本中，Debezium 引入了 JDBC sink 连接器实现，它发现了只专注于为关系和非关系数据库开发源连接器。Debezium JDBC sink 连接器与其他供应商实现不同，因为它能够充分利用 Debezium 连接器发送的原始更改事件，而无需首先应用事件扁平化转换。Debezium JDBC sink 连接器可以利用原生 Debezium 源连接器功能，如列类型传播，允许您降低数据管道的处理空间，并简化其配置。

以下示例显示了一个简单的配置，从名为 orders 的 Kafka 主题评估事件到 PostgreSQL 数据库。主题中的事件由 Debezium MySQL 连接器发出，而无需使用 ExtractNewRecordState 转换。

```
{
 "name": "mysql-to-postgres-pipeline",
 "config": {
  "connector_class": "io.debezium.connector.jdbc.JdbcSinkConnector",
  "topics": "orders",
  "connection.url": "jdbc://postgresql://<host>:<port>/<database>",
  "connection.user": "<username>",
  "connection.password": "<password>",
  "insert.mode": "upsert",
  "delete.enabled": "true",
  "primary.key.mode": "record_key",
```

```
    "schema.evolution": "basic"
  }
}
```

前面的例子显示了一系列 connection<.> 属性，用于定义用于访问目标 PostgreSQL 数据库的连接字符串和凭证。在写入目标数据库时，记录使用 *UPSERT* 语义，使用插入来创建记录（如果不存在），或者更新记录（如果确实存在）。启用 schema evolution，表的键列派生自事件的主键。

您可以将 JDBC sink 连接器发行版本与以下关系数据库搭配使用：

- Db2
- MySQL
- Oracle
- PostgreSQL
- SQL Server

如需更多信息，请参阅 JDBC 的 Debezium 连接器。

2.5.5.2. MySQL 连接器：并行快照（开发者预览）

Debezium 初始快照进程始终是关系数据库的单线程。这种限制主要源自确保多个交易间的数据一致性的复杂性。

从这个版本开始，您可以在执行一致的数据库快照时将连接器配置为使用多个线程。这种实现使用这些线程并行执行表级快照。

要利用并行快照，请在连接器配置中设置 snapshot.max.threads 属性，并为它分配一个大于 1 的值。

使用并行快照的配置示例

```
snapshot.max.threads=4
```

根据前面的示例，连接器会并行快照最多 4 个表。如果有多个快照表，在一个线程完成后，连接器会处理队列中的下一个表。此过程将继续，直到所有表都是快照。

2.5.5.3. Oracle 连接器：从 Oracle 逻辑待机中获得更改（开发者预览）

Oracle 的 Debezium 连接器维护一个内部 *flush 表*，用于监控 Oracle Log Writer Buffer (LGWR)进程的清除周期。连接器访问数据库的用户帐户必须具有创建并写入 flush 表的权限。对数据操作的逻辑支持通常具有更严格的规则，甚至是只读的，因此写入数据库是不良的，甚至不可遗漏。

要启用连接器从 Oracle 只读逻辑数据库进行最大更改，本发行版本引入了一个标志，用于禁用此 *flush 表的* 创建和管理。您可以将此开发人员预览功能用于 Oracle Standalone 和 Oracle RAC 安装。

要启用连接器从 Oracle 只读逻辑独立进行最接近的更改，请添加以下连接器选项：

```
internal.log.mining.read.only=true
```

2.5.5.4. PostgreSQL 连接器的一次交付（开发者预览）

通常，Debezium 一直是一次交付解决方案，确保不会错过任何更改。一次交付是作为 KIP-618 的一部分 Apache Kafka 社区的提议。本提议旨在解决生产者（源连接器）在重试过程中遇到的常见问题。连接器可能会将批处理事件重新发送到 Kafka 代理，即使代理已经提交了批处理。这种情况可能会导致发送重复的事件，这可能会导致无法轻松处理重复的用户（接收器连接器）出现问题。

不需要更改连接器配置，才能利用完全运行一次的交付。但是，要启用精确发送，您必须调整 Kafka Connect worker 配置以使用 KIP-618 中引入的配置属性。在 Debezium 2.3.4 中，PostgreSQL 的精确语义仅在流传输阶段应用，而不是在快照期间应用。

2.5.6. 本发行版本中的其他更新

此 Debezium 2.3.4 发行版本提供多个功能更新和修复，包括以下列表中的项目：

- DBZ-1973 启用 Debezium 来发送有关其状态的通知
- DBZ-2296 更好的控制 Debezium GTID 使用情况
- DBZ-2979 Connector 在更改为排除列后发出事件记录
- DBZ-3594 使用 snapshot.collection.include.list 时，关系架构没有被正确填充
- DBZ-4027 make signalling 频道可配置
- DBZ-4488 Failed retriable 操作会无限重试
- DBZ-4663 Remove 选项用于从 MySQL 连接器指定驱动程序类
- MySQL 文档中不存在 DBZ-4829 Property event.processing.failure.handling.mode
- DBZ-5282 Debezium 无法使用 Apicurio 和自定义信任存储
- DBZ-5283 Add 选项，以排除 ExtractNewRecordState SMT 中未更改的字段
- 在启用了 LOB 时， DBZ-5395 Connector offsets 不会提前在带有过滤的事件进行事务提交中。
- DBZ-5490 Document message.key.columns and tombstone events limitations for default REPLICA IDENTITY
- DBZ-5798 Data type conversion failed for MySQL BIGINT
- DBZ-5917 Unable 用于指定列或表，如果名称包含反斜杠\，则包含列表。

- DBZ-5879 支持在连接器启动过程中重试数据库连接失败
- DBZ-5907 Oracle 无法撤销更改
- 重启时 DBZ-5915 PostgreSQL 数据丢失
- DBZ-5945 Oracle 多线程丢失数据
- DBZ-5966 Truncate 记录与 ExtractNewRecordState 不兼容
- DBZ-5967 Computed 分区不能为负数
- DBZ-5973 MongoDB incremental 快照无法正常工作
- DBZ-5985 Table size 日志消息用于 snapshot.select.statement.overrides 表不正确
- 使用 exclude.tables 配置对指定错误的表名称执行快照信号中的 DBZ-5988 NPE
- DBZ-5991 在 PostgreSQL 连接器解析金级的边界值存在问题
- DBZ-5993 Log statement for unparseable DDL statement in MySqlDatabaseSchema 包含占位符
- DBZ-6001 PostgreSQL 连接器将资金类型的 null 解析为 0
- DBZ-6003 Nullable 列在 DDL 事件中标有"可选：false"
- DBZ-6012 PostgreSQL LSN 检查应遵循 event.processing.failure.handling.mode

- 在 PostgreSQL 连接器上遇到错误时，DBZ-6026 Offsets 不会在连接偏移主题中清除
- DBZ-6029 Unexpected format for TIME column: 8:00
- DBZ-6031 Oracle 不支持 LOB 存储条款后的 compression/logging 子句
- DBZ-6037 Debezium 将记录完整消息以及错误
- DBZ-6039 在从 Kafka 恢复内部模式历史记录的过程中提高了弹性
- DBZ-6046 执行 PostgreSQL 数据库升级时添加 Debezium 步骤
- DBZ-6051 Incremental 快照将事件从信号数据库发送到 Kafka
- DBZ-6064 Mask password in log statement
- DBZ-6075 Loading custom offset storage fails with Class not found 错误
- DBZ-6079 increases query.fetch.size 默认为 sensible zero
- 如果数据库数量小于 maxTasks，则 DBZ-6084 SQL Server 任务会失败
- DBZ-6089 Expose sequence field in CloudEvents message id
- 如果事务没有与捕获的表相关的事件，则 DBZ-6094 Reduce 跳过的事务的详细程度
- DBZ-6107 在使用 LOB 支持时，针对多行的 UPDATE 可能会导致事件数据不一致

- DBZ-6112 PostgreSQL：连接器启动时设置 Replica 身份
- 当处理不同的字符数组和日期数组时，DBZ-6122 PostgreSQL 连接器会失败
- DBZ-6131 支持使用 MongoDB 的聚合管道步骤更改流过滤
- DBZ-6219 突出显示有关如何配置 schema 历史记录主题的信息，以仅存储预期表的数据
- DBZ-6254 Introduce LogMiner 查询过滤模式
- 当部署多个连接器时，DBZ-6256 Lock contention on LOG_MINING_FLUSH 表
- DBZ-6329 在 Oracle 更改事件源信息块中 rs_id 字段为 null
- DBZ-6353 使用 pg_replication_slot_advance，PostgreSQL10 不支持。
- DBZ-6355 log.mining.transaction.retention.hours 应该引用最后一个偏移而不是 sysdate
- DBZ-6366 Code Improvements for skip.messages.without.change
- DBZ-6379 Toasted hstore 没有被正确处理
- DBZ-6386 Oracle DDL 缩小空间无法解析
- DBZ-6396 PostgreSQL connector 任务无法恢复流，因为复制插槽处于活跃状态
- DBZ-6402 MongoDB 连接器在无效的恢复令牌中崩溃

- DBZ-6439 在快照期间，Oracle 连接器需要很长时间才能读取捕获的表的结构
- 使用多租户时，DBZ-6457 Oracle parallel 快照没有正确设置 PDB 上下文
- DBZ-6459 [MariaDB] 添加了对 userstat 插件关键字的支持
- DBZ-6474 Oracle snapshot.include.collection.list 应使用文档中的 databaseName 前缀。
- DBZ-6485 Db2 连接器在通知发送时可能会失败
- DBZ-6486 ExtractNewRecordState SMT 与 HeaderToValue SMT 相结合会导致 Unexpected 字段名称异常
- 当队列内存大小限制处于位时，DBZ-6490 BigDecimal 会失败
- DBZ-6492 Oracle 表无法捕获，获得 runtime.NoViableAltException
- DBZ-6496 Signal poll interval 具有不正确的默认值
- DBZ-6502 Oracle JDBC 驱动程序 23.x 会抛出 ORA-18716 - 不在任何时区
- DBZ-6509 FileSignalChannel 没有加载
- DBZ-6512 Debezium 增量快照块大小文档不明确或不正确
- DBZ-6513 Error value of negative seconds in convertoracleIntervalDaySecond
- DBZ-6515 Debezium 增量快照块大小文档不明确或不正确

- DBZ-6524 [PostgreSQL] LTree 数据没有被流捕获
- DBZ-6528 Oracle Connector: Snapshot 使用特定组合失败
- DBZ-6529 使用更好的分区哈希功能
- DBZ-6533 Table order 在快照中不正确
- DBZ-6543 Unhandled NullPointerException in PartitionRouting 会使整个连接插件崩溃
- DBZ-6559 Bug in field.name.adjustment.mode 属性
- DBZ-6585 Oracle 不支持的 DDL 语句 - 丢弃多个分区
- DBZ-6589 支持 UUID、JSON 和 JSONB 数据类型的 PostgreSQL 协调
- DBZ-6590 MySQL parser 无法解析 CAST AS dec
- DBZ-6599 Oracle DDL parser 在注释模糊处理分号时无法正确检测声明结束
- DBZ-6605 修复为表扫描完成通知的 DataCollections
- 如果 ORA-01327 被另一个 JDBC 或 Oracle 异常嵌套，则 DBZ-6610 Oracle 连接器无法恢复
- 在解析 MySQL 时 DBZ-6613 Fatal 错误(Percona 5.7.39-42)流程
- DBZ-6622 MySQL ALTER USER 带有 RETAIN CURRENT PASSWORD 失败，但解析异常

- DBZ-6628 Inaccurate 文档中有关 额外条件的文档
- DBZ-6633 Oracle 连接 SQLRecoverableExceptions 默认不会被重试
- DBZ-6643 MongoDB 连接器会保持上线。通过 DBZ-6670 修复
- DBZ-6648 Cannot delete non-null interval 值
- DBZ-6670 Retriable 操作会无限重试，因为错误处理程序不会被重复使用
- DBZ-6677 Oracle DDL parser 不支持 ALTER TABLE 上的列可见性
- DBZ-6690 Should 在 MBean 命名中使用 topic.prefix 而不是 connector.server.name
- DBZ-6716 Oracle 无法处理 DROP USER
- DBZ-6724 Debezium 在解析 MySQL DDL 语句时崩溃（特定的 JOIN）
- DBZ-6725 ExtractNewDocumentState for MongoDB 在处理删除事件时忽略以前的文档状态
- DBZ-6733 Oracle LogMiner mining distance 计算，当上限没有距离时，应该跳过
- DBZ-6736 MariaDB: Unparseable DDL 语句(ALTER TABLE IF EXISTS)
- DBZ-6758 当在 postgres 连接器中使用 pgoutput 时，(+/-) Infinity 不支持十进制值
- DBZ-6760 Outbox 转换可能会导致连接器崩溃

- DBZ-6774 MongoDB New Document State Extraction: non existing field for add.headers
- 使用 JMX 频道时，DBZ-6777 通知和信号泄漏 MBean 实例
- DBZ-6780 Debezium 在解析 MySQL DDL 语句时崩溃(SELECT 1.;)
- DBZ-6794 Debezium 在解析 MySQL DDL 语句时崩溃(SELECT 1 + @sum:=1 AS s;)
- DBZ-6803 MySQL 连接器异常，因为 DDL 解析器不接受 REPEAT 功能
- 当 DDL 语句声明包含非拉丁字符的变量名称时，DBZ-6821 Debezium 会崩溃
- DBZ-6824 在解析 MySQL DDL 时，连接器现在可以正确修剪 BIGINT 和 SMALLINT 类型的默认值
- DBZ-6830 Partial 和 multi-response 事务现在只记录在 debug 模式中
- DBZ-6867 Streaming 聚合管道中断，用于数据库过滤器和信号集合的组合

2.6. 已弃用的功能

本发行版本中已弃用以下功能：

mongodb.hosts 属性不再被支持。要将集成连接器配置为连接到 MongoDB 副本集，请使用 mongodb.connection.string 属性。

2.7. 已知问题

以下已知问题会影响 Debezium 2.3.4：

- **Apicurio registry 2.4.3 和 2.4.4 会导致 Kafka 上的无限重新平衡循环**

第 3 章 SERVICE REGISTRY 2.5 发行注记

Service Registry 是标准事件模式和 API 设计的数据存储，它基于 Apicurio Registry 开源社区项目。

注意

红帽构建的 Apicurio Registry 现在作为 Red Hat Application Foundations 的一部分提供。红帽构建的 Apicurio Registry 2.x 和 Red Hat Integration Service Registry 2.x 的功能相同。如需更多信息，请参阅 Red Hat Application Foundation。

您可以使用 Service Registry 通过 Web 控制台、REST API、Maven 插件或 Java 客户端管理和共享数据的结构。例如，客户端应用程序可以动态地向 Service Registry 或从 Service Registry 推送最新的架构更新，而无需重新部署。您还可以创建可选规则来管理 Service Registry 内容随时间变化的方式。这些规则包括验证内容、工件引用的完整性，以及向后兼容或转发 schema 或 API 版本的兼容性。

3.1. SERVICE REGISTRY 安装选项

您可以使用以下数据存储选项之一在 OpenShift 上安装 Service Registry：

- PostgreSQL 数据库
- Red Hat AMQ Streams

如需了解更多详细信息，请参阅在 OpenShift 上安装和部署 Service Registry。

3.2. SERVICE REGISTRY 支持的平台

Service Registry 2.5 支持以下核心平台：

- Red Hat OpenShift Container Platform: 4.15, 4.14, 4.13, 4.12
- Red Hat OpenShift Service on AWS: 4.13

- Microsoft Azure Red Hat OpenShift: 4.13
- PostgreSQL: 15, 14, 13, 12
- Red Hat AMQ Streams: 2.6, 2.5, 2.2
- OpenJDK: 17, 11

如需了解更多详细信息，请参阅以下文章：

- Red Hat Integration Service Registry 支持的配置。

3.2.1. 支持的与其他产品集成

Service Registry 2.5 还支持与以下产品集成：

- Red Hat Single Sign-On (RH-SSO) 7.6
- 红帽构建的 Debezium 2.3

3.2.2. Operator 元数据版本

有关用于安装和部署 Service Registry 的对应 Service Registry Operator 元数据版本的详情，请查看以下文章：

- Red Hat Integration - Service Registry Operator 元数据版本。

3.3. SERVICE REGISTRY 新功能

Service Registry 2.5 包括以下新功能：

Service Registry 核心新功能

升级到 Quarkus 3.x

- Service Registry 服务器运行时已从 Quarkus 2.x 升级到 Quarkus 3.x。此升级提供了更高的安全性、性能和维护。如需了解更多详细信息，请参阅 https://quarkus.io/quarkus3/。Service Registry 2.5 基于 Quarkus 3.2 构建。

avro SerDes 改进

- 在使用 Apache Avro serializers/deserializers 时，支持生成带有 null 字段的模式。如需了解更多详细信息，请参阅 Registry-3862。

模式缓存容错

- 添加了选项以使用现有模式缓存条目，而不是在 schema 缓存加载失败时抛出错误。如需了解更多详细信息，请参阅 Registry-3807。

解引用工件内容

- 在某些情况下，返回带有引用的内容的工件内容可能会很有用。在这些情况下，Core Registry API v2 添加了对某些操作中 dereference 查询参数的支持。如需了解更多详细信息，请参阅 Apicurio Registry v2 core REST API 文档。

- 目前，当 API 操作中指定 dereference 参数时，这个支持只对 Avro 和 Protobuf 工件实现。这个参数不支持任何其他工件类型。如需了解更多详细信息，请参阅 Registry-2865。

注意

对于 Protobuf 工件，只有在所有模式都属于同一软件包时才支持取消引用内容。

Service Registry Maven 插件改进

- 添加 选项，以跳过 Maven 插件中的注册目标。如需了解更多详细信息，请参阅 Registry-3817。

- 使用 pom.xml 文件中的 autoRef 选项，自动检测 Maven 插件中的引用。如需了解更多详细信息，请参阅 Registry-3439。这是一个技术预览功能。

重要

技术预览功能不受红帽产品服务等级协议（SLA）支持，且功能可能并不完整。红帽不推荐在生产环境中使用它们。这些技术预览功能可以使用户提早试用新的功能，并有机会在开发阶段提供反馈意见。

有关红帽技术预览功能支持范围的更多信息，请参阅技术预览功能支持范围。

Service Registry Operator 的新功能

改进了对 SQL 数据源配置的支持

- Service Registry Operator 支持使用环境变量配置 SQL 数据源，作为 spec.configuration.sql.dataSource 字段的替代选择。现在，您可以在 ApicurioRegistry 自定义资源中使用 Kubernetes secret 而不是明文提供 SQL 凭证。如需了解更多详细信息，请参阅 https://access.redhat.com/solutions/7059053。

- 这个版本改进了 Service Registry Operator，以更好地支持这个用例。现在，您可以使用 spec.configuration.sql.dataSource 和 spec.configuration.env 字段来定义配置的部分。例如，以下配置现在有效：

```
apiVersion: registry.apicur.io/v1
kind: ApicurioRegistry
metadata:
  name: myregistry
spec:
  configuration:
    persistence: sql
  sql:
    dataSource:
      url: "jdbc:postgresql://..."
      userName: "postgres-user"
    env:
      - name: REGISTRY_DATASOURCE_PASSWORD
        valueFrom:
          secretKeyRef:
            name: postgres-secret
            key: password
```

Operator 还检测到这种类型的配置，并在无需其他用户干预的情况下立即应用。

Service Registry 用户文档和示例

文档库已使用版本 2.5 中的新功能更新：

- 在 OpenShift 上安装和部署 Service Registry
- 迁移 Service Registry 部署
- Service Registry 用户指南
- Apicurio Registry v2 core REST API 文档

开源演示应用程序也已更新：

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

3.4. SERVICE REGISTRY 弃用的功能

Service Registry 核心已弃用的功能

- *Confluent Schema Registry API 版本 6 (compatibility API)*: Service Registry 目前支持独立端点上的 Confluent Schema Registry API 的两个版本：版本 6 和版本 7。v6 API 端点已弃用，并将在以后的发行版本中删除。确保将对 v6 API 端点的所有引用替换为对 v7 API 端点的引用。
- *Service Registry Core API 版本 1*: Service Registry Core API 的原始版本 1 的支持现已弃用。此 v1 旧 API 将在下一个主发行版本中删除。
- *动态日志级别配置* ： v2 Service Registry Core API 中弃用了 /admin/loggers 和 /admin/loggers/{logger} API 端点。这些端点将在以后的发行版本中被删除。
- *Registry V1 export 工具* ：命令行导出工具的 Service Registry 支持现已弃用。导出工具用于将 Service Registry 1.x 中的数据导出为可导入到 2.x 的格式，将不再发布或维护。所有客户都应该已从 1.x 升级到 2.x。

Service Registry Operator 已弃用的功能

- *JAVA_OPTIONS 环境变量* ： JAVA_OPTIONS 环境变量不再是为 Service Registry 配置 Java 选项的首选方法。您可以使用 JAVA_OPTS_APPEND 环境变量。JAVA_OPTS 环境变量也可用，它取代了 Java 选项的默认内容。但是，最好避免使用 JAVA_OPTS，因为它可能会影响一些 Service Registry Operator 功能。

- *通过编辑 Deployment 资源来设置环境变量* ：在以前的版本中，您可以通过直接编辑其 Deployment 资源（由 Service Registry Operator 支持）来为 Service Registry 设置环境变量。现在，您可以使用 ApicurioRegistry CRD 文件中的 spec.configuration.env 字段来管理环境变量，前面的流程已弃用，并将删除对它的 Operator 支持。确保使用 spec.configuration.env 字段来设置 Operator 未设置的所有环境变量。

- *为未启用的功能保留环境变量* ： Service Registry Operator 设置环境变量以启用和配置各种功能，如使用 Kafka 存储时 Salted Challenge Response Authentication Mechanism (SCRAM)安全性。当禁用此类功能时，Operator 当前会保留关联的环境变量，这可能会导致问题。此类环境变量的保留已弃用，Operator 对它的支持将被删除。确保您的部署不依赖于这些环境变量的保留。

- *环境变量优先级* ： Service Registry Operator 可能会尝试设置已在 spec.configuration.env 字段中明确指定的环境变量。如果环境变量具有冲突的值，则 Service Registry Operator 设置的值会默认具有优先权。此行为将在以后改变，以便用户覆盖 Operator 设置的大部分环境变量。确保您的部署不依赖于原始优先级行为。

3.5. 升级和迁移 SERVICE REGISTRY 部署

您可以将 Service Registry 服务器从 Service Registry 2.x 升级到 OpenShift 上的 Service Registry 2.5。没有从 Service Registry 1.x 到 Service Registry 2.x 的自动升级，需要迁移过程。

3.5.1. 更新 2.x 客户端依赖项

这不是更新本发行版本的客户端依赖项所必需的。现有 Service Registry 2.x 客户端应用程序将继续用于 Service Registry 2.5。

但是，在下一个 Service Registry 版本前，您必须更新所有客户端依赖项以使用最新版本的 Service Registry。客户端依赖项包括 Service Registry Kafka serializers/deserializers (SerDes)、Maven 插件和 Java 客户端应用程序的依赖项。

例如，要更新 Java 客户端应用程序的 Maven 依赖项，请在 pom.xml 文件中指定版本，如下所示：

```
<dependency>
```

```
    <groupId>io.apicurio</groupId>
    <artifactId>apicurio-registry-client</artifactId>
    <version>2.5.10.Final-redhat-00001</version>
</dependency>
```

如需了解更多详细信息，请参阅 默认启用旧 REST API 日期格式。

3.5.2. 从 OpenShift 上的 Service Registry 2.x 升级

您可以从 OpenShift 4.11 上的 Service Registry 2.x 升级到 OpenShift 4.12 或更高版本的 Service Registry 2.5。您必须升级 Service Registry 和 OpenShift 版本，并一次升级 OpenShift 的一个次版本。

前提条件

- 您已在 OpenShift 4.11 或更高版本上安装了 Service Registry 2.x。
- 您已在 Kafka 主题或 PostgreSQL 数据库中备份了现有 Service Registry 存储数据。如需了解更多详细信息，请参阅在 OpenShift 上安装和部署 Service Registry。

重要

在 OpenShift 上的生产环境中，为了帮助确保在升级前备份存储，最好将 Service Registry 的 Operator 更新批准策略设置为 manual 而不是 automatic。

流程

1. 在 OpenShift Container Platform Web 控制台中，点 Administration，然后点 Cluster Settings。
2. 点 Channel 字段旁边的铅笔图标，然后选择下一个次 candidate 版本（例如，从 stable-4.11 改为 candidate-4.12）。
3. 点 Save 然后点 Update，等待升级完成。
4. 如果 OpenShift 版本小于 4.13，请重复步骤 2 和 3，然后选择 candidate-4.13 或更高版本。

5. 点 Operators > Installed Operators > Red Hat Integration - Service Registry。

6. 确保 更新频道 已设置为 2.x。

7. 如果 Update approval 设为 Automatic，则升级应在设置 2.x 频道后立即批准并安装。

8. 如果 Update approval 设置为 Manual，点 Install。

9. 等待 Operator 部署好并部署了 Service Registry pod。

10. 验证您的 Service Registry 系统是否正在运行。

其他资源

- 有关如何在 OpenShift Container Platform Web 控制台中设置 Operator 更新频道的更多详细信息，请参阅 更改 Operator 的更新频道。

3.5.3. 从 OpenShift 上的 Service Registry 1.1 迁移

有关从 Service Registry 1.1 迁移到 Service Registry 2.x 的详情，请参阅 迁移 Service Registry 部署。

3.6. SERVICE REGISTRY 解决的问题

表 3.1. 解决了 Service Registry 2.5.10 中的问题

问题	描述
IPT-1091	Service Registry Operator 应该支持 JAVA_OPTS_APPEND 和 JAVA_OPTIONS （已弃用）环境变量。
IPT-1092	Service Registry 升级会破坏 Confluent v6 兼容性 API。

表 3.2. 解决了 Service Registry 2.5.9 中的问题

问题	描述
IPT-1071	使用 KafkaSQL 存储升级 Service Registry 时可能的数据丢失，以及带有引用的 Protobuf 工件。
IPT-1035	Operator 升级到 2.2.3 后 CrashLoop 中的 Service Registry Pod。
registry-4417	无法从 Service Registry 中正确删除孤立的内容。
registry-4283	当为单个工件创建具有相同名称的两个引用时，Service Registry 服务器应该会失败。
registry-4226	删除所有规则 REST API 操作不会删除 **INTEGRITY** 规则（仅限KafkaSQL 存储）。
registry-4215	带有不同字段顺序的 avro 以规范形式被视为相等。
Registry-4107	以只读模式在 Apicurio Registry web 控制台中不会显示有效性、兼容性和完整性规则值。

表 3.3. Service Registry 2.5.5 中解决的问题

问题	描述
Registry-4104	当配置了 AMQ Streams 存储和 OAuth 时，Service Registry 可能会因为缺少的 kafka-oauth-client 类而无法启动。

表 3.4. 解决了 Service Registry 2.5.4 中的问题

问题	描述
registry-4019	即使计数器到达限制，一些健康检查也始终为 UP。
Registry-3956	即使在本地缓存中已存在 schema (SerDes)，也会调用 schema registry。
Registry-3725	资源所有者密码 grant - 基本身份验证 - java.lang.IllegalStateException: Client is closed.
Registry-3647	protobuf 内容规范过时的值被检测到。

3.7. SERVICE REGISTRY 解决了 CVE

以下常见漏洞和暴露(CVE)在 Service Registry 2.5 中解决：

表 3.5. Apicurio Registry 2.5.9 中解决的 CVE

CVE	描述
CVE-2024-20952 CVE-2024-20921 CVE-2024-20919 CVE-2024-20918	很难利用漏洞，攻击者可以通过多个协议访问网络，从而破坏了用于 JDK、Oracle GraalVM Enterprise Edition 的 Oracle Java SE、Oracle GraalVM Enterprise Edition 的 Oracle Java SE、Oracle GraalVM Enterprise Edition。
CVE-2024-20945	很难利用漏洞，在日志中对 Oracle Java SE、Oracle GraalVM for JDK、Oracle GraalVM Enterprise Edition 执行的低特权攻击者、Oracle GraalVM Enterprise Edition 对 JDK、Oracle GraalVM Enterprise Edition 的 Oracle Java SE、Oracle GraalVM Enterprise Edition 造成影响。
CVE-2024-20932	易利用的漏洞可让通过多个协议进行网络访问的未经身份验证的攻击者破坏了用于 JDK、Oracle GraalVM Enterprise Edition 的 Oracle Java SE、Oracle GraalVM Enterprise Edition。
CVE-2023-39615	在 Libxml2 中发现了一个安全漏洞，它在 /libxml2/SAXX2.c 的 xmlSAX2StartElement （） 函数中包含全局缓冲区溢出。
CVE-2023-38473	Avahi 中发现了一个漏洞。avahi_alternative_host_name （）函数中存在可访问断言。
CVE-2023-38472	Avahi 中发现了一个漏洞。avahi_rdata_parse （）函数中存在可访问断言。
CVE-2023-38471	Avahi 中发现了一个漏洞。dbus_set_host_name 函数中存在可访问断言。
CVE-2023-38470	Avahi 中发现了一个漏洞。avahi_escape_label （）函数中存在可访问断言。
CVE-2023-38469	Avahi 中发现了一个漏洞，它在 avahi_dns_packet_append_record 中存在一个可访问的断言。
CVE-2023-27043	通过 3.11.3 的 Python 的电子邮件模块错误地解析包含特殊字符的电子邮件地址。
CVE-2023-7104	SQLite3 中发现了一个漏洞。此问题会影响 make alltest Handler 组件中的 ext/session/sqlite3session.c 函数的 sessionReadRecord 功能。操作可能会导致基于堆的缓冲区溢出。
CVE-2023-5981	发现了一个漏洞，在 RSA-PSK ClientKeyExchange 中格式密码文本的响应时间与 ciphertexts 的响应时间不同，并带有正确的 PKCS#1 v1.5 padding。
CVE-2023-5678	OpenSSL 中发现了一个安全漏洞，这会导致生成或检查较长的 X9.42 DH 密钥或参数要比预期要慢得多。此问题可能会导致拒绝服务。
CVE-2023-5388	发现，在 NSS 中使用用于 RSA 加密的数字库会泄漏信息，无论 RSA 解密结果的高顺序是零。
CVE-2023-3817 CVE-2023-3446	OpenSSL 中发现了一个漏洞。发生此安全问题的原因是，使用 DH_check （）、DH_check_ex （）或 EVP_PKEY_param_check （）函数的应用程序来检查 DH 密钥或 DH 参数可能会长时间延迟。

CVE	描述
CVE-2022-48564	在 plistlib.py 文件中的 read_ints （）函数中的 Python core plistlib 库中发现了一个漏洞。
CVE-2022-48560	通过 heapq 模块中的 heappushpop 函数在 Python 中发现一个 use-after-free 漏洞。
CVE-2021-3468	在 avahi 中发现了一个安全漏洞。在 avahi Unix 套接字上指示客户端连接终止的事件没有在 client_work 功能中正确处理，允许本地攻击者触发无限循环。

表 3.6. 解决了 Service Registry 2.5.4 中的 CVE 问题

问题	描述
IPT-1034	CVE-2023-5072 JSON-java: parser混淆会导致 OOM 错误。
IPT-1030	CVE-2023-31582 jose4j: Insecure iteration count 设置。
IPT-1021	CVE-2023-44487 undertow: HTTP/2: 启用多个 HTTP/2 的 Web 服务器会受到 DDoS 攻击(Rapid Reset Attack)的影响。
IPT-1013	CVE-2023-39410 avro: apache-avro: Apache Avro Java SDK: Memory when deserialing not data in Avro Java SDK.
IPT-995	CVE-2023-4853 quarkus-vertx-http: quarkus: HTTP 安全策略绕过。
IPT-993	CVE-2023-39321 CVE-2023-39322 integration-service-registry-operator-container: 各种漏洞。
IPT-953	CVE-2023-29409 integration-service-registry-operator-container: golang: crypto/tls: slow 验证包含大型 RSA 密钥的证书链。
IPT-948	CVE-2023-29406 integration-service-registry-operator-container: golang: net/http: insufficient sanitization of Host 标头。
IPT-940	CVE-2023-34462 netty: SniHandler 16MB 分配会导致 OutOfMemoryError。
IPT-936	CVE-2023-34455 snappy-java: 未检查的块长度会导致 DoS。
IPT-935	CVE-2023-35116 jackson-databind: 通过 cyclic 依赖项拒绝服务。
IPT-874	CVE-2023-1584 quarkus-oidc: ID，通过授权代码流访问令牌泄漏。

表 3.7. Apicurio Registry 2.5.4 中解决的额外 CVE

CVE	描述
CVE-2023-44483	所有版本的 Apache Santuario - 2.2.6、2.3.4 和 3.0.3 之前的 Java 的 XML 安全性都会受到使用 JSR 105 API 的问题的影响，当生成带有 debug 级别的 XML 签名和日志记录时，可以在日志文件中披露私钥。
CVE-2023-43642	在 snappy-java 中的 SnappyInputStream 中发现了一个安全漏洞，它是 Java 中的数据压缩库。当因为块长度缺少上限检查而解压缩带有太大的块数据时，会发生此问题。
CVE-2023-42503	Apache Commons Compress: Denial of service via CPU consumption for malformed TAR 文件。
CVE-2023-40217	Python 3 ssl.SSLSocket 容易受到对 HTTPS 服务器在特定实例中绕过 TLS 握手的攻击，以及使用 TLS 客户端身份验证的其他服务器端协议，如 mTLS。
CVE-2021-39194	在解析带有 kaml 中标记的 polymorphism 风格的 polymorphic 输入时拒绝服务
CVE-2023-34454 CVE-2023-34453	在 Snappy-java 的 shuffle 函数中发现了一个安全漏洞，它在开始操作前不会检查输入大小。
CVE-2023-29491	在 ncurses 中发现了一个漏洞，由 setuid 应用程序使用时进行。
CVE-2023-28118	在使用定位符和别名解析输入时，kaml 的潜在拒绝服务。
CVE-2022-24823	当在 netty 中使用多部分解码器时，如果启用了在磁盘上存储上传，则本地信息披露可能会通过本地系统临时目录发生。
CVE-2023-4911	在处理 GLIBC_TUNABLES 环境变量时，GNU C 库的动态加载程序 ld.so 中发现了一个缓冲区溢出。
CVE-2023-4813	glibc 中发现了一个安全漏洞。在不常用的情形中，gaih_inet 函数可以使用已释放的内存，从而导致应用程序崩溃。
CVE-2023-4806	glibc 中发现了一个安全漏洞。在非常罕见的情况下，getaddrinfo 函数可以访问已释放的内存，从而导致应用程序崩溃。
CVE-2023-4527	glibc 中发现了一个安全漏洞。当使用 AF_UNSPEC 地址系列调用 getaddrinfo 函数，且系统通过 /etc/resolv.conf 配置为 no-aaaa 模式时，通过 TCP 大于 2048 字节的 DNS 响应可以通过函数返回的地址数据披露堆栈内容，并可能导致崩溃。

3.8. SERVICE REGISTRY 已知问题

以下已知问题在 Service Registry 2.5 中应用：

Service Registry 内核已知问题

registry-3413 - 默认启用旧 REST API 日期格式

为获得最大兼容性，并从旧版本的 Service Registry 更轻松地升级，Service Registry REST API 中使用的日期格式与 OpenAPI 标准不兼容。这是因为旧版本中的一个错误。

在下一个 Service Registry 版本前，您必须升级所有客户端应用程序以使用最新的 Service Registry 客户端版本。下一个版本将修复日期格式 bug，这会导致旧的客户端不再与 REST API 兼容。

要将 REST API 更新至兼容 OpenAPI，您可以修复此 Service Registry 中的日期格式错误，如下所示：

1. *将所有客户端应用程序更新至 2.5.10 版本。Final-redhat-00001，如 更新 2.x 客户端依赖项中所述。*

2. *将以下环境变量设置为显示的值：*

```
REGISTRY_APIS_V2_DATE_FORMAT=yyyy-MM-dd'T'HH:mm:ss'Z'
```

IPT-814 - Service Registry logout 功能与 RH-SSO 7.6 不兼容

在 RH-SSO 7.6 中，与 logout 端点一起使用的 redirect_uri 参数已弃用。如需了解更多详细信息，请参阅 RH-SSO 7.6 升级指南。因此，当 Service Registry 使用 RH-SSO Operator 保护时，点 Logout 按钮会显示 Invalid parameter: redirect_uri 错误。

有关临时解决方案，请参阅 https://access.redhat.com/solutions/6980926。

IPT-701 - CVE-2022-23221 H2 允许通过 JNDI 从远程服务器加载自定义类

当 Service Registry 数据存储在 AMQ Streams 中时，H2 数据库控制台允许远程攻击者使用 JDBC URL 执行任意代码。在默认情况下，Service Registry 不会受到攻击，需要进行恶意配置更改。

Service Registry Operator 已知问题

operator-42 - 自动生成 OpenShift 路由可能会使用错误的基本主机值

如果指定了多个 routerCanonicalHostname 值，则自动生成 Service Registry OpenShift 路由可能会使用错误的基本主机值。

附录 A. 使用您的订阅

集成通过软件订阅提供。要管理您的订阅，请访问红帽客户门户中的帐户。

访问您的帐户

1. 转至 access.redhat.com。
2. 如果您还没有帐户，请创建一个帐户。
3. 登录到您的帐户。

激活订阅

1. 转至 access.redhat.com。
2. 导航到 My Subscriptions。
3. 导航到 激活订阅 并输入您的 16 位激活号。

下载 ZIP 和 TAR 文件

要访问 zip 或 tar 文件，请使用客户门户网站查找下载的相关文件。如果您使用 RPM 软件包，则不需要这一步。

1. 打开浏览器并登录红帽客户门户网站 产品下载页面，网址为 access.redhat.com/downloads。
2. 向下滚动到 INTEGRATION 和 AUTOMATION。
3. 点 Red Hat Integration 以显示 Red Hat Integration 下载页面。

4. *单击组件的 **Download** 链接。*

更新于 2024-03-23