



Red Hat Integration 2021.Q1

Debezium の OpenShift へのインストール

OpenShift Container Platform での Debezium 1.4 の使用

Red Hat Integration 2021.Q1 Debezium の OpenShift へのインストール

OpenShift Container Platform での Debezium 1.4 の使用

Enter your first name here. Enter your surname here.

Enter your organisation's name here. Enter your organisational division here.

Enter your email address here.

法律上の通知

Copyright © 2022 | You need to change the HOLDER entity in the en-US/Installing_Debezium_on_OpenShift.ent file |.

The text of and illustrations in this document are licensed by Red Hat under a Creative Commons Attribution–Share Alike 3.0 Unported license ("CC-BY-SA"). An explanation of CC-BY-SA is available at

<http://creativecommons.org/licenses/by-sa/3.0/>

. In accordance with CC-BY-SA, if you distribute this document or an adaptation of it, you must provide the URL for the original version.

Red Hat, as the licensor of this document, waives the right to enforce, and agrees not to assert, Section 4d of CC-BY-SA to the fullest extent permitted by applicable law.

Red Hat, Red Hat Enterprise Linux, the Shadowman logo, the Red Hat logo, JBoss, OpenShift, Fedora, the Infinity logo, and RHCE are trademarks of Red Hat, Inc., registered in the United States and other countries.

Linux[®] is the registered trademark of Linus Torvalds in the United States and other countries.

Java[®] is a registered trademark of Oracle and/or its affiliates.

XFS[®] is a trademark of Silicon Graphics International Corp. or its subsidiaries in the United States and/or other countries.

MySQL[®] is a registered trademark of MySQL AB in the United States, the European Union and other countries.

Node.js[®] is an official trademark of Joyent. Red Hat is not formally related to or endorsed by the official Joyent Node.js open source or commercial project.

The OpenStack[®] Word Mark and OpenStack logo are either registered trademarks/service marks or trademarks/service marks of the OpenStack Foundation, in the United States and other countries and are used with the OpenStack Foundation's permission. We are not affiliated with, endorsed or sponsored by the OpenStack Foundation, or the OpenStack community.

All other trademarks are the property of their respective owners.

概要

本ガイドでは、AMQ Streams を使用して OpenShift Container Platform に Red Hat Debezium をインストールする方法を説明します。

目次

はじめに	3
多様性を受け入れるオープンソースの強化	3
第1章 DEBEZIUM の概要	4
第2章 DEBEZIUM コネクターのインストール	5
2.1. 前提条件	5
2.2. KAFKA トピック作成に関する推奨事項	5
2.3. AMQ STREAMS での DEBEZIUM のデプロイ	6
付録A サブスクリプションの使用	9
アカウントへのアクセス	9
サブスクリプションのアクティベート	9
zip および tar ファイルのダウンロード	9

はじめに

多様性を受け入れるオープンソースの強化

Red Hat では、コード、ドキュメント、Web プロパティにおける配慮に欠ける用語の置き換えに取り組んでいます。まずは、マスター (master)、スレーブ (slave)、ブラックリスト (blacklist)、ホワイトリスト (whitelist) の 4 つの用語の置き換えから始めます。この取り組みは膨大な作業を要するため、今後の複数のリリースで段階的に用語の置き換えを実施して参ります。詳細は、[Red Hat CTO である Chris Wright のメッセージ](#) をご覧ください。

第1章 DEBEZIUM の概要

Red Hat Debezium は、データベース操作をキャプチャーし、行レベル操作のデータ変更イベントレコードを作成して、Kafka トピックに変更イベントレコードをストリーミングする分散型プラットフォームです。Red Hat Debezium は Apache Karaf に構築され、AMQ Streams とデプロイおよび統合されます。

Debezium によって、データベーステーブルの行レベルの変更がキャプチャーされ、対応する変更イベントが AMQ Streams に渡されます。アプリケーションはこれらの **変更イベントストリーム** を読み取りでき、発生順に変更イベントにアクセスできます。

Debezium には、以下を含む複数の用途があります。

- データレプリケーション
- キャッシュの更新およびインデックスの検索
- モノリシックアプリケーションの簡素化
- データ統合
- ストリーミングクエリーの有効化

Debezium は、以下の共通データベースのコネクター (Kafka Connect をベースとする) を提供します。

- MySQL
- PostgreSQL
- SQL Server
- MongoDB
- Db2



重要

Debezium Db2 コネクターはテクノロジープレビュー機能です。テクノロジープレビューの機能は、Red Hat の実稼働環境のサービスレベルアグリーメント (SLA) ではサポートされず、機能的に完全ではないことがあるため、Red Hat はテクノロジープレビュー機能を実稼働環境に実装することは推奨しません。テクノロジープレビューの機能は、最新の技術をいち早く提供して、開発段階で機能のテストやフィードバックの収集を可能にするために提供されます。サポート範囲の詳細は、[テクノロジープレビュー機能のサポート範囲](#) を参照してください。

[Debezium](#) は、Red Hat Debezium のアップストリームコミュニティプロジェクトです。

第2章 DEBEZIUM コネクターのインストール

コネクタプラグインで Kafka Connect を拡張して、AMQ Streams 経由で Debezium コネクターをインストールします。AMQ Streams のデプロイ後に、Kafka Connect で Debezium をコネクタ設定としてデプロイできます。

2.1. 前提条件

Debezium のインストールには、以下が必要です。

- OpenShift クラスター。
- Kafka Connect での AMQ Streams のデプロイメント。
- 必要なクラスターロールおよび API サービスを設定するための **cluster-admin** 権限を持つ OpenShift クラスターのユーザー。



注記

Debezium コネクターを実行するには、Java 8 以降が必要です。

Debezium をインストールするには、OpenShift Container Platform コマンドラインインターフェイス (CLI) が必要です。OpenShift 4.7 の CLI のインストール方法については、[OpenShift Container Platform 4.7 のドキュメント](#) を参照してください。

その他のリソース

- AMQ Streams のインストール方法については、[AMQ Streams on OpenShift の使用](#) を参照してください。
- AMQ Streams には、Kafka コンポーネントをデプロイおよび管理する **Cluster Operator** が含まれています。AMQ Streams Cluster Operator を使用して Kafka コンポーネントをインストールする方法の詳細は、[Kafka Connect のクラスターへのデプロイ](#) を参照してください。

2.2. KAFKA トピック作成に関する推奨事項

Debezium は、データの保存に複数の Kafka トピックを使用します。トピックは、管理者が作成するか、[ブローカー設定プロパティ `auto.create.topics.enable`](#) を使用してトピックの自動作成を有効にし、Kafka 自身が作成する必要があります。

以下のリストで、トピックの作成時に考慮すべき制限および推奨事項を説明します。

MySQL、SQL Server、および Db2 コネクターのデータベース履歴トピック

- 無限または非常に長期の保持期間
- 3 以上の実稼働でのレプリケーション係数
- 単一パーティション

その他のトピック

- 特定のレコードの **最後の** 変更イベントのみを保持するために [Kafka ログの圧縮](#) が有効になっている場合は、Apache Kafka で **min.compaction.lag.ms** および **delete.retention.ms**

トピックレベルの設定を設定します。コンシューマーがすべてのイベントを受信し、マーカを削除するために十分な時間を確保する必要があります。よって、シンクコネクタで予想される最大ダウンタイムよりも大きな値を設定します。たとえば、コネクタ更新時のダウンタイムを考慮します。

- 実稼働でレプリケートされます。
- 単一パーティション。
単一パーティションルールを緩和できますが、アプリケーションはデータベースの異なる行に対して順不同のイベントを処理する必要があります。単一行のイベントは、引き続き完全に順序付けされます。複数のパーティションを使用する場合、Kafka がキーをハッシュ化してパーティションを決定するのがデフォルトの動作になります。その他のパーティションストラテジーでは、SMT (Simple Message Transform、シンプルメッセージ変換) を使用して各レコードにパーティション番号を設定する必要があります。

2.3. AMQ STREAMS での DEBEZIUM のデプロイ

Red Hat OpenShift Container Platform で Debezium コネクタを設定するには、Kafka クラスターを OpenShift にデプロイし、Debezium コネクタをダウンロードおよび設定して、コネクタで Kafka Connect をデプロイします。

前提条件

- [Red Hat AMQ Streams](#) を使用して OpenShift で Apache Kafka および Kafka Connect が設定済みである。AMQ Streams は、Kafka を OpenShift に取り入れる operator およびイメージを提供します。
- Podman または Docker がインストールされている。

手順

1. Kafka クラスターをデプロイします。Kafka クラスターがすでにデプロイされている場合は、以下の3つのサブステップを省略します。
 - a. [AMQ Streams のインストールおよびコンポーネントのデプロイ](#) の手順にしたがって、AMQ Streams operator をインストールします。
 - b. 希望の設定を選択し、[Kafka Cluster をデプロイ](#) します。
 - c. [Kafka Connect](#) をデプロイします。

これで、Kafka Connect を使用して OpenShift で実行されている Kafka クラスターを利用できます。

2. Pod が稼働していることを確認します。Pod 名は AMQ Streams デプロイメントに対応します。

```
$ oc get pods
```

```
NAME                                READY STATUS
<cluster-name>-entity-operator-7b6b9d4c5f-k7b92  3/3 Running
<cluster-name>-kafka-0                    2/2 Running
<cluster-name>-zookeeper-0                2/2 Running
<cluster-name>-operator-97cd5cf7b-l58bq      1/1 Running
```

Pod の実行の他に、Kafka Connect に関連付けられた **DeploymentConfig** が必要です。

3. [Red Hat Integration のダウンロードサイト](#) に移動します。
4. データベースの Debezium コネクタアーカイブをダウンロードします。
5. アーカイブを展開して、コネクタプラグインのディレクトリ構造を作成します。複数のアーカイブをダウンロードおよび展開した場合、構造は以下のようになります。

```
$ tree ./my-plugins/
./my-plugins/
├── debezium-connector-db2
│   └── ...
├── debezium-connector-mongodb
│   └── ...
├── debezium-connector-mysql
│   └── ...
├── debezium-connector-postgres
│   └── ...
└── debezium-connector-sqlserver
    └── ...
```

6. **registry.redhat.io/amq7/amq-streams-kafka-26-rhel7:1.6.0** をベースイメージとして使用して、新規の **Dockerfile** を作成します。

```
FROM registry.redhat.io/amq7/amq-streams-kafka-26-rhel7:1.6.0
USER root:root
COPY ./my-plugins/ /opt/kafka/plugins/
USER 1001
```

7. コンテナイメージをビルドします。前の手順で作成した **Dockerfile** がカレントディレクトリにある場合は、以下のコマンドのいずれかを入力します。

```
podman build -t my-new-container-image:latest .
```

```
docker build -t my-new-container-image:latest .
```

8. カスタムイメージをコンテナレジストリーにプッシュします。以下のいずれかのコマンドを実行します。

```
podman push my-new-container-image:latest
```

```
docker push my-new-container-image:latest
```

9. 新しいコンテナイメージを示します。Debezium コネクタを実行するために作成したイメージの名前を指定するため、以下のいずれかのタスクを実行します。

- **KafkaConnect** カスタムリソースの **spec.image** フィールドを編集します。
このプロパティを設定すると、値は Cluster Operator の **STRIMZI_DEFAULT_KAFKA_CONNECT_IMAGE** 変数を上書きします。以下に例を示します。

```
apiVersion: kafka.strimzi.io/v1beta1
```

```
kind: KafkaConnect
metadata:
  name: my-connect-cluster
  annotations: strimzi.io/use-connector-resources: "true"
spec:
  #...
  image: my-new-container-image
```

- **install/cluster-operator/050-Deployment-strimzi-cluster-operator.yaml** ファイルの **STRIMZI_DEFAULT_KAFKA_CONNECT_IMAGE** 変数を編集し、新しいコンテナイメージを示すようにした後、Cluster Operator を再インストールします。このファイルを編集する場合は、これを OpenShift クラスターに適用する必要があります。

Kafka Connect デプロイメントによって、新しいイメージの使用が開始されます。

次のステップ

- デプロイする各 Debezium コネクタに、コネクタインスタンスを設定する **KafkaConnect** カスタムリソースを作成し、適用します。これにより、設定されたデータベースに対してコネクタの実行が開始されます。コネクタが起動すると、設定されたデータベースに接続し、挿入、更新、および削除された各行または各ドキュメントの変更イベントレコードを生成します。コネクタのデプロイに関する詳細は、以下を参照してください。
 - [Deploying the MySQL connector](#)
 - [Deploying the MongoDB connector](#)
 - [Deploying the PostgreSQL connector](#)
 - [Deploying the SQL Server connector](#)
 - [Deploying the Db2 connector](#)
Db2 コネクタを使用するには、IBM InfoSphere Data Replication (IIDR) 製品のライセンスが必要です。ただし、IIDR をインストールする必要はありません。
- **KafkaConnect.spec.image property** および **STRIMZI_DEFAULT_KAFKA_CONNECT_IMAGE** 変数の詳細は、[Using AMQ Streams on OpenShift](#) を参照してください。

付録A サブスクリプションの使用

AMQ Streams は、ソフトウェアサブスクリプションによって提供されます。サブスクリプションを管理するには、Red Hat カスタマーポータルでアカウントにアクセスします。

アカウントへのアクセス

1. access.redhat.com に移動します。
2. アカウントがない場合は、作成します。
3. アカウントにログインします。

サブスクリプションのアクティベート

1. access.redhat.com に移動します。
2. **サブスクリプション** に移動します。
3. **Activate a subscription** に移動し、16 桁のアクティベーション番号を入力します。

zip および tar ファイルのダウンロード

zip または tar ファイルにアクセスするには、カスタマーポータルを使用して、ダウンロードする関連ファイルを検索します。RPM パッケージを使用している場合は、この手順は必要ありません。

1. ブラウザーを開き、access.redhat.com/downloads で Red Hat カスタマーポータルの **Product Downloads** ページにログインします。
2. **INTEGRATION AND AUTOMATION** まで下方向にスクロールします。
3. **Red Hat Integration** をクリックして、Red Hat Integration ダウンロードページを表示します。
4. コンポーネントの **ダウンロード** リンクをクリックします。

改訂日時: 2022-12-03 12:06:54 +1000