



Red Hat AMQ Streams 2.2

OpenShift での AMQ Streams の使用開始

OpenShift Container Platform での AMQ Streams 2.2 の使用開始

Red Hat AMQ Streams 2.2 OpenShift での AMQ Streams の使用開始

OpenShift Container Platform での AMQ Streams 2.2 の使用開始

法律上の通知

Copyright © 2023 Red Hat, Inc.

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

OpenShift で Kafka クラスターを作成して、AMQ Streams を試してください。Kafka クラスターに接続し、Kafka トピックからメッセージを送受信します。

目次

多様性を受け入れるオープンソースの強化	3
第1章 概要	4
1.1. OPERATORHUB の使用	4
1.2. 前提条件	4
1.3. 関連情報	5
第2章 OPERATORHUB からの AMQ STREAMS OPERATOR のインストール	6
第3章 AMQ STREAMS OPERATOR を使用した KAFKA コンポーネントのデプロイ	8
第4章 KAFKA クラスターにアクセスするための OPENSIFT ルートの作成	9
第5章 トピックからのメッセージの送受信	12
付録A サブスクリプションの使用	15
アカウントへのアクセス	15
サブスクリプションのアクティベート	15
Zip および Tar ファイルのダウンロード	15
DNF を使用したパッケージのインストール	15

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 概要

Red Hat AMQ Streams を使用して Kafka クラスターの作成およびセットアップを行ってから、アプリケーションとサービスをそれらのクラスターに接続します。

このガイドでは、AMQ Streams on OpenShift Container Platform をインストールして使い始める方法について説明します。AMQ Streams Operator は、OpenShift Web コンソールの OperatorHub からインストールできます。この AMQ Streams Operator は、Kafka コンポーネントのインストールおよび管理方法を把握しています。OperatorHub からインストールすると、自動更新を利用できる AMQ Streams の標準設定が提供されます。

AMQ Streams Operator がインストールされると、Kafka コンポーネントのインスタンスをインストールするためのリソースが提供されます。Kafka クラスターをインストールした後、メッセージの生成と消費を開始できます。



注記

デプロイメントにさらに柔軟性が必要な場合は、AMQ Streams で提供されるインストールアーティファクトを使用できます。インストールアーティファクトの使用の詳細については、[OpenShift での AMQ Streams のデプロイおよびアップグレード](#)を参照してください。

1.1. OPERATORHUB の使用

OperatorHub は、Operator をインストールするための OpenShift Container Platform の Web コンソールインターフェイスです。OperatorHub を使用すると、OpenShift Web コンソールから AMQ Streams Operator を直接インストールできます。

次のいずれかの方法で AMQ Streams Operator をインストールすることを選択できます。

AMQ Streams Operator のみ

[AMQ Streams Operator を直接インストールします。](#)

Red Hat Integration Operator (非推奨)

複数の Red Hat Integration サブスクリプションがある場合、Red Hat Integration Operator を使用して、AMQ Streams Operator およびサブスクライブしている Red Hat Integration コンポーネントのすべての Operator をインストールおよび更新できます。詳細については、[Red Hat Integration Operator のインストール](#)を参照してください。



注記

Red Hat Integration Operator は非推奨となり、今後削除される予定です。OpenShift 4.6 から 4.10 では、OperatorHub で利用できます。

1.2. 前提条件

AMQ Streams を使い始めるには、次の前提条件を満たす必要があります。

- Red Hat アカウントを持っている。
- JDK 11 以降がインストールされている。
- OpenShift 4.10 ~ 4.14 クラスターを使用できる。

- OpenShift **oc** コマンドラインツールがインストールされ、稼働中のクラスターに接続するように設定されている。

開始する手順は、OpenShift Web コンソールの OperatorHub の使用に基づいていますが、**OpenShift oc** CLI ツールを使用して特定の操作を実行することもできます。**oc** ツールを使用して OpenShift クラスターに接続する必要があります。

- '?' ヘルプメニュー、**Command Line Tools**の順にクリックすると、Web コンソールから **oc** CLI ツールをインストールできます。
- プロファイル名をクリックしてから **Copy login command** をクリックすると、Web コンソールから必要な **oc login** の詳細をコピーできます。

1.3. 関連情報

- [Strimzi の概要](#)
- [AMQ Streams on OpenShift のデプロイおよびアップグレード](#)
- [Red Hat Integration Operator のインストール](#)

第2章 OPERATORHUB からの AMQ STREAMS OPERATOR のインストール

OpenShift Container Platform Web コンソールの OperatorHub を使用して、AMQ Streams Operator をインストールしてサブスクライブできます。

この手順では、プロジェクトを作成し、そのプロジェクトに AMQ Streams Operator をインストールする方法を説明します。プロジェクトは namespace を表します。namespace を使用して機能を分離することで管理性を確保することを推奨します。



警告

適切な更新チャンネルを使用するようにしてください。サポート対象の OpenShift のバージョンを使用している場合は、デフォルトの stable チャンネルから安全に AMQ Streams をインストールできます。ただし、stable チャンネルで自動更新を有効にすることは推奨されません。自動アップグレードでは、アップグレード前の必要手順がスキップされます。バージョン固有のチャンネルでのみ自動アップグレードを使用します。

前提条件

- **cluster-admin** または **strimzi-admin** パーミッションを持つアカウントを使用して OpenShift Container Platform Web コンソールにアクセスできる。

手順

1. OpenShift Web コンソールで **Home > Projects** ページに移動し、インストール用のプロジェクト (namespace) を作成します。
この例では、**amq-streams-kafka** という名前のプロジェクトを使用します。
2. **Operators > OperatorHub** ページに移動します。
3. スクロール、または **Filter by keyword** ボックスにキーワードを入力して、**Red Hat Integration - AMQ Streams Operator** を見つけます。
Operator は、**Streaming & Messaging** カテゴリにあります。
4. **Red Hat Integration - AMQ Streams** をクリックして、Operator 情報を表示します。
5. Operator に関する情報を確認し、**Install** をクリックします。
6. **Install Operator** ページで、次のインストールおよび更新オプションから選択します。
 - **Update Channel:** Operator の更新チャンネルを選択します。
 - **stable** チャンネル (デフォルト) には最新の更新とリリースがすべて含まれます。これには、十分なテストを行った上、安定していることが想定される、メジャー、マイナー、およびマイクロリリースが含まれます。
 - **amq-streams-X.x** チャンネルには、メジャーリリースのマイナーリリースの更新およびマイクロリリースの更新が含まれます。X は、メジャーリリースのバージョン番号に置き換えてください。

- `amq-streams-X.Y.x` チャネルには、マイナーリリースのマイクロリリースの更新が含まれます。X はメジャーリリースのバージョン番号、Y はマイナーリリースのバージョン番号に置き換えてください。
 - **Installation Mode:** 作成したプロジェクトを選択して、特定の namespace に Operator をインストールします。
AMQ Streams Operator をクラスターのすべての namespace (デフォルトのオプション) にインストールするか、特定の namespace にインストールするかを選択できます。特定の namespace を Kafka クラスターおよびその他の AMQ Streams コンポーネント専用とすることが推奨されます。
 - **Update approval:** デフォルトでは、OLM (Operator Lifecycle Manager) によって、AMQ Streams Operator が自動的に最新の AMQ Streams バージョンにアップグレードされます。今後のアップグレードを手動で承認する場合は、**Manual** を選択します。詳細は、OpenShift ドキュメントの [Operators](#) ガイドを参照してください。
7. **Install** をクリックして、選択した namespace に Operator をインストールします。
AMQ Streams Operator によって、Cluster Operator、CRD、およびロールベースアクセス制御 (RBAC) リソースは選択された namespace にデプロイされます。
 8. Operator を使用する準備ができたなら、**Operators > Installed Operators** に移動して、Operator が選択した namespace にインストールされていることを確認します。
ステータスは **Succeeded** と表示されます。

これで、AMQ Streams Operator を使用して、Kafka クラスターから順に Kafka コンポーネントをデプロイできるようになりました。



注記

Workloads > Deployments に移動すると、Cluster Operator および Entity Operator のデプロイメントの詳細を確認できます。Cluster Operator の名前には、バージョン番号 **amq-streams-cluster-operator-<version>** が含まれます。AMQ Streams インストールアーティファクトを使用して Cluster Operator をデプロイする場合、名前は異なります。この場合、名前は **strimzi-cluster-operator** です。

第3章 AMQ STREAMS OPERATOR を使用した KAFKA コンポーネントのデプロイ

Openshift にインストールすると、AMQ Streams Operator は、ユーザーインターフェイスから Kafka コンポーネントをインストールできるようにします。

次の Kafka コンポーネントをインストールできます。

- Kafka
- Kafka Connect
- Kafka MirrorMaker
- Kafka MirrorMaker 2
- Kafka Topic
- Kafka User
- Kafka Bridge
- Kafka Connector
- Kafka Rebalance

コンポーネントを選択して、インスタンスを作成します。少なくとも、Kafka インスタンスを作成します。この手順では、デフォルト設定を使用して Kafka インスタンスを作成する方法を説明します。インストールを実行する前に、デフォルトのインストール仕様を設定できます。

プロセスは、他の Kafka コンポーネントのインスタンスを作成する場合と同じです。

前提条件

- AMQ Streams Operator が [OpenShift クラスタにインストールされている](#)。

手順

1. Web コンソールで **Operators > Installed Operators** ページに移動し、**Red Hat Integration - AMQ Streams** をクリックして、Operator の詳細を表示します。
提供されている API から、Kafka コンポーネントのインスタンスを作成できます。
2. **Kafka** の下の **Create instance** をクリックして、Kafka インスタンスを作成します。
デフォルトでは、3つの Kafka ブローカーノードと3つの ZooKeeper ノードを持つ **my-cluster** という名の Kafka クラスタを作成します。クラスタはエフェメラルストレージを使用します。
3. **Create** をクリックして、Kafka のインストールを開始します。
ステータスが **Ready** に変わるまで待ちます。

第4章 KAFKA クラスターにアクセスするための OPENSIFT ルートの作成

OpenShift の外部で Kafka クラスターにアクセスするための OpenShift ルートを作成します。

この手順では、Kafka クラスターを OpenShift 環境外のクライアントに公開する方法を説明します。Kafka クラスターが公開された後、外部クライアントは Kafka クラスターからのメッセージを生成および消費できます。

OpenShift ルートを作成するために、OpenShift にインストールされている Kafka クラスターの設定に **route** リスナーが追加されます。



警告

OpenShift Route アドレスは、Kafka クラスターの名前、リスナーの名前、および作成される namespace の名前で設定されます。たとえば、**my-cluster-kafka-listener1-bootstrap-amq-streams-kafka** (`<cluster_name>-kafka-<listener_name>-bootstrap-<namespace>`) です。アドレスの全体の長さが上限の 63 文字を超えないように注意してください。

前提条件

- [OpenShift で Kafka クラスターを作成](#) している。
- 証明書を管理するには、OpenJDK **keytool** が必要である。
- (オプション) OpenShift **oc** CLI ツールを使用していくつかのステップを実行できる。

手順

1. Web コンソールで **Operators > Installed Operators** ページに移動し、**Red Hat Integration - AMQ Streams** を選択して、Operator の詳細を表示します。
2. **Kafka** ページを選択して、インストールされている Kafka クラスターを表示します。
3. 設定している Kafka クラスターの名前をクリックして、その詳細を表示します。この例では、**my-cluster** という名前の Kafka クラスターを使用します。
4. Kafka クラスター **my-cluster** の **YAML** ページを選択します。
5. ルートリスナー設定を追加して、**listener1** という名前の OpenShift ルートを作成します。リスナー設定は、**route** タイプに設定する必要があります。Kafka 設定の **listeners** の下にリスナー設定を追加します。

外部ルートリスナーの設定

```
apiVersion: kafka.strimzi.io/v1beta2
kind: Kafka
metadata:
  name: my-cluster
```

```

namespace: amq-streams-kafka
spec:
  kafka:
    # ...
    listeners:
      # ...
      - name: listener1
        port: 9094
        type: route
        tls: true
    # ...

```

クライアントはデフォルトのルーターポートであるポート 443 に接続しますが、トラフィックは設定するポート (この例では 9094) にルーティングされます。

- 更新された設定を保存します。
- Kafka クラスター **my-cluster** の **Resources** ページを選択して、クライアントに必要な接続情報を見つけます。
Resources ページから、Kafka クラスターに接続するために必要なルートリスナーと公開クラスター証明書の詳細を確認できます。
- Kafka クラスター用に作成された **my-cluster-kafka-listener1-bootstrap** ルートの名前をクリックして、ルートの詳細を表示します。
- ホスト名をメモします。
ホスト名は、Kafka クラスターに接続するためのブートストラップアドレスとして、Kafka クライアントのポート 443 で指定されます。

Networking > **Routes** に移動し、**amq-streams-kafka** プロジェクトを選択して、namespace に作成されたルートを表示することにより、ブートストラップアドレスを見つけることもできます。

または、**oc** ツールを使用してブートストラップの詳細を抽出できます。

ブートストラップ情報の抽出

```
oc get routes my-cluster-kafka-listener1-bootstrap -o=jsonpath='{.status.ingress[0].host}'{"\n"}
```

- Resources** ページに戻り、**my-cluster-cluster-ca-cert** の名前をクリックして、Kafka クラスターにアクセスするためのシークレットの詳細を表示します。
ca.crt 証明書ファイルには、Kafka クラスターの公開証明書が含まれています。

Kafka ブローカーにアクセスするには証明書が必要です。

- ca.crt** 公開証明書ファイルのローカルコピーを作成します。
証明書の詳細をコピーするか、OpenShift **oc** ツールを使用してそれらを抽出できます。

公開証明書の抽出

```
oc extract secret/my-cluster-cluster-ca-cert --keys=ca.crt --to=- > ca.crt
```

- keytool** を使用して、公開クラスター証明書のローカルトラストストアを作成します。

ローカルトラストストアの作成

```
keytool -keystore client.truststore.jks -alias CARoot -import -file ca.crt
```

プロンプトが表示されたら、トラストストアにアクセスするためのパスワードを作成します。

トラストストアは、Kafka クラスターへのアクセスを認証するために Kafka クライアントで指定されます。

メッセージの送受信を開始する準備が整いました。

第5章 トピックからのメッセージの送受信

OpenShift にインストールされている Kafka クラスターとの間でメッセージを送受信します。

この手順では、Kafka クライアントを使用してメッセージを生成および消費する方法を説明します。クライアントを OpenShift にデプロイするか、ローカル Kafka クライアントを OpenShift クラスターに接続できます。いずれかまたは両方のオプションを使用して、Kafka クラスターのインストールをテストできます。ローカルクライアントの場合は、OpenShift ルート接続を使用して Kafka クラスターにアクセスします。

oc コマンドラインツールを使用して、Kafka クライアントをデプロイして実行します。

前提条件

- [OpenShift で Kafka クラスターを作成](#) している。

ローカルプロデューサーおよびコンシューマーの場合:

- [OpenShift で実行している Kafka クラスターへの外部アクセス用のルートを作成](#) している。
- [AMQ Streams ソフトウェアダウンロードページ](#) から最新の Kafka クライアントバイナリーにアクセスできる。

OpenShift クラスターにデプロイされた Kafka クライアントからのメッセージの送受信

プロデューサーおよびコンシューマーのクライアントを OpenShift クラスターにデプロイします。その後、クライアントを使用して、同じ namespace 内の Kafka クラスターとの間でメッセージを送受信できます。デプロイメントでは、Kafka を実行するために AMQ Streams コンテナイメージを使用します。

1. **oc** コマンドラインインターフェイスを使用して、Kafka プロデューサーをデプロイします。この例では、Kafka クラスター **my-cluster** に接続する Kafka プロデューサーをデプロイします。

my-topic という名前のトピックが作成されます。

Kafka プロデューサーの OpenShift へのデプロイ

```
oc run kafka-producer -ti \
  --image=registry.redhat.io/amq7/amq-streams-kafka-32-rhel8:2.2.2 \
  --rm=true \
  --restart=Never \
  -- bin/kafka-console-producer.sh \
  --bootstrap-server my-cluster-kafka-bootstrap:9092 \
  --topic my-topic
```

2. コマンドプロンプトから、いくつかのメッセージを入力します。
3. OpenShift Web コンソールで **Home > Projects** ページに移動し、作成した **amq-streams-kafka** プロジェクトを選択します。
4. Pod のリストから、**kafka-producer** をクリックして、プロデューサー Pod の詳細を表示します。
5. **Logs** ページを選択して、入力したメッセージが存在することを確認します。

6. **oc** コマンドラインインターフェイスを使用して、Kafka コンシューマーをデプロイします。

Kafka コンシューマーの OpenShift へのデプロイ

```
oc run kafka-consumer -ti \
--image=registry.redhat.io/amq7/amq-streams-kafka-32-rhel8:2.2.2 \
--rm=true \
--restart=Never \
-- bin/kafka-console-consumer.sh \
--bootstrap-server my-cluster-kafka-bootstrap:9092 \
--topic my-topic \
--from-beginning
```

コンシューマーは **my-topic** に生成されたメッセージを消費しました。

7. コマンドプロンプトから、コンシューマーコンソールに着信メッセージが表示されていることを確認します。
8. OpenShift Web コンソールで **Home > Projects** ページに移動し、作成した **amq-streams-kafka** プロジェクトを選択します。
9. Pod のリストから、**kafka-consumer** をクリックして、コンシューマー Pod の詳細を表示します。
10. **Logs** ページを選択して、消費したメッセージが存在することを確認します。

ローカルで実行されている Kafka クライアントからのメッセージの送受信

コマンドラインインターフェイスを使用して、ローカルマシンで Kafka プロデューサーとコンシューマーを実行します。

1. [AMQ Streams ソフトウェアダウンロードページ](#) から **AMQ Streams<version>** バイナリーをダウンロードして展開します。
amq-streams-<version>-bin.zip ファイルを任意の場所に解凍します。
2. コマンドラインインターフェイスを開き、トピック **my-topic** と TLS の認証プロパティーを使用して Kafka コンソールプロデューサーを起動します。
[OpenShift ルートを使用して Kafka ブローカーにアクセスする](#) のに必要なプロパティーを追加します。
 - 使用している OpenShift ルートのホスト名およびポート 443 を使用します。
 - パスワードと、ブローカー証明書用に作成したトラストストアへの参照を使用します。

ローカル Kafka プロデューサーの起動

```
kafka-console-producer.sh \
--bootstrap-server my-cluster-kafka-listener1-bootstrap-amq-streams-kafka.apps.ci-ln-50kcyvt-72292.origin-ci-int-gce.dev.rhcloud.com:443 \
--producer-property security.protocol=SSL \
--producer-property ssl.truststore.password=password \
--producer-property ssl.truststore.location=client.truststore.jks \
--topic my-topic
```

3. プロデューサーが実行しているコマンドラインインターフェイスにメッセージを入力します。

4. Enter を押してメッセージを送信します。
5. 新しいコマンドラインインターフェイスタブまたはウィンドウを開き、Kafka コンソールコンシューマーを起動してメッセージを受信します。
プロデューサーと同じ接続の詳細を使用します。

ローカル Kafka コンシューマーの起動

```
kafka-console-consumer.sh \  
--bootstrap-server my-cluster-kafka-listener1-bootstrap-amq-streams-kafka.apps.ci-ln-50kcyvt-72292.origin-ci-int-gce.dev.rhcloud.com:443 \  
--consumer-property security.protocol=SSL \  
--consumer-property ssl.truststore.password=password \  
--consumer-property ssl.truststore.location=client.truststore.jks \  
--topic my-topic --from-beginning
```

6. コンシューマーコンソールに受信メッセージが表示されることを確認します。
7. Ctrl+C を押して、Kafka コンソールプロデューサーとコンシューマーを終了します。

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から提供されます。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. **My Subscriptions** 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、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** カテゴリで、**AMQ Streams for Apache Kafka** エントリーを見つけます。
3. 必要な AMQ Streams 製品を選択します。**Software Downloads** ページが開きます。
4. コンポーネントの **Download** リンクをクリックします。

DNF を使用したパッケージのインストール

パッケージとすべてのパッケージ依存関係をインストールするには、以下を使用します。

```
dnf install <package_name>
```

ローカルディレクトリーからダウンロード済みのパッケージをインストールするには、以下を使用します。

```
dnf install <path_to_download_package>
```

改訂日時: 2023-10-30