OpenShift Container Platform 4.8

OpenShift 向けのサンドボックスコンテナーのサポート

OpenShift サンドボックスコンテナーガイド

Last Updated: 2023-04-18

OpenShift Container Platform 4.8 OpenShift 向けのサンドボックスコンテナーのサポート

OpenShift サンドボックスコンテナーガイド

法律上の通知

概要

OpenShift Container Platform の OpenShift サンドボックスコンテナーがサポートされることで、追加のオプションランタイムとして、Kata Container を実行するビルドインサポートが追加されます。

目次

第1章 {SANDBOXED-CONTAINERS-FIRST} 1.0 リリースノート

1.1. 本リリースについて

これらのリリースノートは、Red Hat OpenShift Container Platform での OpenShift サンドボックスコンテナーの開発を追跡します。

この製品は現在テクノロジープレビューとして提供されています。OpenShift サンドボックスコンテナーは、実稼働環境での使用を目的としていません。詳細は、テクノロジープレビューの機能に関する Red Hat カスタマーポータルの サポート範囲 を参照してください。

1.2. 新機能および機能拡張

1.2.1. OpenShift Container Platform での OpenShift サンドボックスコンテナーのサポート (テクノロジープレビュー)

OpenShift サンドボックスコンテナー 1.0.0 テクノロジープレビューリリースでは、追加のランタイムとして Kata コンテナーを実行するための組み込みサポートが導入されています。OpenShift サンドボックスコンテナーを使用すると、ユーザーは追加のランタイムとして Kata コンテナーを選択して、ワークロードをさらに分離できます。OpenShift サンドボックスコンテナー Operator は、Kata コンテナーのインストール、削除、および更新のタスクを自動化します。**KataConfig** カスタムリソースを記述することにより、これらのタスクの状態を追跡できます。

OpenShift サンドボックスコンテナーは、ベアメタルでのみサポートされます。Red Hat Enterprise Linux CoreOS (RHCOS) は、OpenShift サンドボックスコンテナー 1.0.0 で唯一サポートされているオペレーティングシステムです。非接続環境は、OpenShift Container Platform 4.8 ではサポートされていません。

詳細は、OpenShift サンドボックスコンテナーについて を参照してください。

1.3. 既知の問題

- OpenShift サンドボックスコンテナーを使用している場合、OpenShift Container Platform クラスターの **hostPath** ボリュームを使用して、ホストノードのファイルシステムから Pod にファイルまたはディレクトリーをマウントすることはできません。別の方法として、ローカル永続ボリュームを使用できます。詳細は 、ローカルボリュームを使用した永続ストレージ を参照してください。(BZ#1904609)

- OpenShift サンドボックスコンテナーで Fedora を実行している場合は、いくつかのパッケージをインストールするための回避策が必要です。**iputils** などの一部のパッケージでは、OpenShift Container Platform がデフォルトでコンテナーに付与しないファイルアクセス許可の変更が必要です。このような特別なアクセス許可を必要とするコンテナーを実行するには、ワークロードを説明するアノテーションを YAML ファイルに追加する必要があります。これは、そのワークロードに対してこのようなファイルのアクセス許可を受け入れるように **virtiofsd** に指示します。必要なアノテーションは以下のとおりです。

  ```
  io.katacontainers.config.hypervisor.virtio_fs_extra_args: |
    [ "-o", "modcaps=+sys_admin", "-o", "xattr" ]
  ```

  (BZ#1915377)

- 4.8 リリースでは、OpenShift Container Platform Web コンソールを使用して **kataConfgPoolSelector** に値を追加すると、**scheduling.nodeSelector** が空の値で設定される

原因となります。**kata** の値で **RuntimeClass** を使用する Pod は、Kata コンテナーランタイムがインストールされていないノードにスケジュールされる場合があります。
この問題を回避するには、以下のコマンドを実行して、**RuntimeClass kata** に手動で **nodeSelector** 値を指定します。

```
$ oc edit runtimeclass kata
```

以下は、正しい **nodeSelector** ステートメントを持つ **RuntimeClass** の例です。

```
apiVersion: node.k8s.io/v1
handler: kata
kind: RuntimeClass
metadata:
  creationTimestamp: "2021-06-14T12:54:19Z"
  name: kata
overhead:
  podFixed:
    cpu: 250m
    memory: 350Mi
scheduling:
  nodeSelector:
    custom-kata-pool: "true"
```

(BZ#2019384)

- Operator Hub の OpenShift サンドボックスコンテナー Operator の詳細ページでは、いくつかのフィールドがありません。フィールドがない場合でも、4.8 で OpenShift サンドボックスコンテナー Operator をインストールできます。(BZ#2019383)

- 複数の **KataConfig** カスタムリソースを作成すると、警告なしで失敗します。OpenShift Container Platform Web コンソールは、複数のカスタムリソースの作成が失敗したことをユーザーに通知するプロンプトを提供しません。(BZ#2019381)

- OpenShift Container Platform Web コンソールの Operator Hub に、Operator のアイコンが表示されない場合があります。(BZ#2019380)

1.4. エラータの非同期更新

OpenShift サンドボックスコンテナー 1.0 のセキュリティー、バグ修正、および機能強化の更新は、Red Hat Network を通じて非同期エラータとして発表されます。OpenShift Container Platform 4.8 のすべてのエラータは Red Hat カスタマーポータルから入手 できます。非同期エラータについては、OpenShift Container Platform ライフサイクル を参照してください。

Red Hat カスタマーポータルのユーザーは、Red Hat Subscription Management (RHSM) のアカウント設定でエラータの通知を有効にすることができます。エラータの通知を有効にすると、登録しているシステムに関連するエラータが新たに発表されるたびに、メールで通知が送信されます。

注記

OpenShift Container Platform のエラータ通知メールを生成させるには、Red Hat カスタマーポータルのユーザーアカウントでシステムが登録されており、OpenShift Container Platform エンタイトルメントを使用している必要があります。

以下のセクションは、これからも継続して更新され、今後の OpenShift sandboxed containers 1.0.0 の非同期リリースで発表されるエラータの拡張機能およびバグ修正に関する情報を追加していきます。

1.4.1. RHBA-2021:3751 - OpenShift サンドボックスコンテナー 1.0.2 バグ修正アドバイザリー

発行日: 2021-10-07

OpenShift サンドボックスコンテナーリリース 1.0.2 が利用可能になりました。このアドバイザリーには、バグ修正を含む OpenShift サンドボックスコンテナーの更新が含まれています。

更新に含まれるバグ修正の一覧は、RHBA-2021:3751 アドバイザリーに記載されています。

1.4.2. RHBA-2021:3552 - OpenShift サンドボックスコンテナー 1.0.1 バグ修正アドバイザリー

発行日: 2021-09-16

OpenShift サンドボックスコンテナーリリース 1.0.1 が利用可能になりました。このアドバイザリーには、バグ修正を含む OpenShift サンドボックスコンテナーの更新が含まれています。

更新に含まれるバグ修正のリストは、RHBA-2021:3552 アドバイザリーに記載されています。

1.4.3. RHEA-2021:2546 - OpenShift サンドボックスコンテナー 1.0.0 イメージのリリース、バグ修正、機能強化のアドバイザリー

発行日: 2021-07-29

OpenShift Container Platform 4.8 の OpenShift サンドボックスコンテナーリリース 1.0.0 サポートのコンポーネントが、テクノロジープレビューとして利用できるようになりました。

更新に含まれるバグ修正の一覧は、RHEA-2021:3941 アドバイザリーに記載されています。

第2章 OPENSHIFT サンドボックスコンテナーについて

OpenShift Container Platform の OpenShift サンドボックスコンテナーがサポートされることで、追加のオプションランタイムとして、Kata Container を実行するビルドインサポートが追加されます。これは、以下のタスクを実行するユーザーに特に便利です。

- 特権または信頼できないワークロードを実行する。
- 各ワークロードのカーネルを確実に分離する。
- テナント全体で同じワークロードを共有する。
- ソフトウェアのテストに適した分離とサンドボックスがあることを確認する。
- 仮想マシン境界を使用して、デフォルトのリソースに含まれるようにする。

OpenShift サンドボックスコンテナーには、さまざまなユースケースに対応するために実行するワークロードのタイプから選択する機能も含まれます。

OpenShift サンドボックスコンテナー Operator を使用して、インストール、削除、更新、ステータスの監視などのタスクを実行できます。

サンドボックスコンテナーは、ベアメタルでのみサポートされます。

Red Hat Enterprise Linux CoreOS (RHCOS) は、OpenShift サンドボックスコンテナー 1.0.0 で唯一サポートされているオペレーティングシステムです。

2.1. OPENSHIFT サンドボックスコンテナーの一般的な用語

以下の用語は、本書全体で使用されています。

サンドボックス

サンドボックスとは、プログラムが実行可能な分離された環境のことです。サンドボックスでは、ホストマシンやオペレーティングシステムに悪影響を及ぼすことなく、テストされていないプログラムまたは信頼できないプログラムを実行できます。
OpenShift サンドボックスコンテナーのコンテキストでは、仮想化を使用して異なるカーネルでワークロードを実行し、同じホストで実行される複数のワークロードとの間の対話を強化することでサンドボックスを実現します。

Pod

Pod は Kubernetes および OpenShift Container Platform から継承されるコンストラクトです。Pod とは、コンテナーのデプロイが可能なリソースを表します。コンテナーは Pod 内で実行され、Pod を使用して複数のコンテナー間で共有できるリソースを指定します。
OpenShift サンドボックスコンテナーのコンテキストでは、Pod が仮想マシンとして実装されます。同じ仮想マシンにある同じ Pod でコンテナーを複数実行できます。

OpenShift サンドボックスコンテナー Operator

Operator は、人間のオペレーターがシステムで実行できるアクション、つまり操作を自動化するソフトウェアコンポーネントです。
OpenShift サンドボックスコンテナー Operator は、クラスター上でサンドボックスコンテナーのライフサイクルを管理してタスクを実行します。これは、サンドボックスコンテナーソフトウェアおよびステータスの監視などの操作を処理します。

Kata Container

Kata Container は OpenShift サンドボックスコンテナーの構築に使用されるコアアップストリームプロジェクトです。OpenShift サンドボックスコンテナーは Kata Container と OpenShift Container Platform を統合します。

KataConfig

KataConfig オブジェクトはサンドボックスコンテナーの設定を表します。ソフトウェアのデプロイ先のノードなど、クラスターの状態に関する情報を保存します。

RHCOS 拡張機能

Red Hat Enterprise Linux CoreOS (RHCOS) 拡張機能は、オプションの OpenShift Container Platform ソフトウェアをインストールするメカニズムです。OpenShift サンドボックスコンテナー Operator はこのメカニズムを使用して、サンドボックスコンテナーをクラスターにデプロイします。

ランタイムクラス

RuntimeClass オブジェクトは、指定のワークロード実行に使用可能なランタイムを記述します。**kata** という名前のランタイムクラスは、OpenShift のサンドボックスコンテナー Operator によってインストールされ、デプロイされます。ランタイムクラスには、ランタイムが Pod オーバーヘッド など、動作に必要なリソースを記述するランタイムに関する情報が含まれます。

2.2. OPENSHIFT サンドボックスコンテナーのビルディングブロック

OpenShift サンドボックス化されたコンテナー Operator は、Kata Container からのコンポーネントをすべてカプセル化します。インストール、ライフサイクル、設定タスクを管理します。

OpenShift サンドボックスコンテナー Operator は、2 つのコンテナーイメージとして Operator バンドル形式 でパッケージ化されます。バンドルイメージにはメタデータが含まれ、Operator で OLM が利用できるようにする必要があります。2 つ目のコンテナーイメージには、**KataConfig** リソースを監視および管理するための実際のコントローラーが含まれています。

2.3. RHCOS 拡張機能

OpenShift サンドボックスコンテナー Operator は Red Hat Enterprise Linux CoreOS (RHCOS) 拡張機能の概念に基づいています。サンドボックスコンテナーの RHCOS 拡張には、Kata、QEMU、およびその依存関係の RPM が含まれます。これらは、Machine Config Operator が提供する **MachineConfig** リソースを使用して有効にできます。

関連情報

- 拡張機能の RHCOS への追加

第3章 OPENSHIFT サンドボックスコンテナーワークロードのデプロイ

Web コンソールまたは OpenShift CLI (**oc**) のいずれかを使用して OpenShift サンドボックスコンテナー Operator をインストールできます。OpenShift サンドボックスコンテナー Operator をインストールする前に、OpenShift Container Platform クラスターを準備する必要があります。

3.1. OPENSHIFT サンドボックスコンテナー向けのクラスターの準備

OpenShift サンドボックスコンテナーをインストールする前に、OpenShift Container Platform クラスターが以下の要件を満たしていることを確認してください。

- クラスターは、Red Hat Enterprise Linux CoreOS (RHCOS) ワーカーを使用してベアメタルインフラストラクチャーにインストールする必要があります。お使いのクラスターではインストーラーでプロビジョニングを使用する必要があります。

重要

- OpenShift サンドボックスコンテナーは RHCOS ワーカーノードのみをサポートします。RHEL 7 ノードまたは RHEL 8 ノードはサポートされません。
- ネストされた仮想化はサポートされていません。

3.1.1. OpenShift サンドボックスコンテナーの追加のリソース要件

OpenShift サンドボックスコンテナーは、Kata Containers などのサンドボックス化されたランタイム内でワークロードを OpenShift Container Platform クラスターで実行できる製品です。各 Pod は仮想マシン (VM) で表されます。各仮想マシンは **qemu** プロセスで実行され、コンテナーワークロードおよびこれらのコンテナーで実行されているプロセスを管理するためのスーパーバイザーとして機能する **kata-agent** プロセスをホストします。オーバーヘッドが増えるプロセスは他に 2 つあります。

- **containerd-shim-kata-v2**。これは Pod との通信に使用されます。
- **virtiofsd**。これはゲストの代わりにホストファイルシステムのアクセスを処理します。

各仮想マシンには、デフォルトのメモリー容量が設定されます。コンテナーでメモリーが明示的に要求された場合に、メモリーが追加で仮想マシンにホットプラグされます。

- コンテナーが指定のメモリーリソースなしで実行される場合は、空きメモリーを使用できます。これは、仮想マシンで使用される合計メモリーがデフォルトの割り当てに到達するまで行われます。ゲストやその I/O バッファーもメモリーを消費します。
- コンテナーに特定のメモリー量が指定されている場合には、コンテナーが起動する前に、メモリーが仮想マシンにホットプラグされます。
- メモリー制限が指定されている場合には、上限より多くメモリーが消費された場合に、ワークロードが終了します。メモリー制限を指定しないと、仮想マシンで実行しているカーネルのメモリーが不足する可能性があります。カーネルでのメモリーが不足すると、仮想マシン上の他のプロセスが終了される可能性があります。

デフォルトのメモリーサイズ

以下の表は、リソース割り当てのデフォルト値を示しています。

リソース	値
デフォルトで仮想マシンに割り当てられるメモリー	2Gi
起動時のゲスト Linux カーネルのメモリー使用量	~110Mi
QEMU プロセスで使用されるメモリー (仮想マシンメモリーを除く)	~30Mi
virtiofsd プロセスで使用されるメモリー (VM I/O バッファーを除く)	~10Mi
containerd-shim-kata-v2 プロセスで使用されるメモリー	~20Mi
Fedora で **dnf install** を実行した後のファイルバッファーのキャッシュデータ	~300Mi* [1]

1. ファイルバッファーが表示され、このバッファーは以下の複数の場所に考慮されます。
   - ファイルバッファーキャッシュとして表示されるゲスト。
   - 許可されたユーザー空間ファイルの I/O 操作をマッピングする **virtiofsd** デーモン。
   - ゲストメモリーとして使用される QEMU プロセス。

注記

メモリー使用量の合計は、メモリー使用率メトリクスによって適切に考慮され、そのメモリーを 1 回だけカウントします。

Pod のオーバーヘッド では、ノード上の Pod が使用するシステムリソースの量を記述します。以下のように、**oc describe runtimeclass kata** を使用して、**kata** ランタイムクラスの現在の Pod オーバーヘッドを取得できます。

例

```
$ oc describe runtimeclass kata
```

出力例

```
Name:         kata
[...]
Metadata:
[...]
Overhead:
  Pod Fixed:
    Cpu:     250m
    Memory:  350Mi
[...]
```

RuntimeClass の **spec.overhead** フィールドを変更して、Pod のオーバーヘッドを変更できます。たとえば、コンテナーに対する設定が QEMU プロセスおよびゲストカーネルデータでメモリー 350Mi 以上を消費する場合に、**RuntimeClass** のオーバーヘッドをニーズに合わせて変更できます。

注記

Red Hat では、指定のデフォルトオーバーヘッド値がサポートされます。デフォルトのオーバーヘッド値の変更はサポートされておらず、値を変更すると技術的な問題が発生する可能性があります。

例

```
kind: RuntimeClass
apiVersion: node.k8s.io/v1
metadata:
  name: kata
overhead:
  podFixed:
    memory: "500Mi"
    cpu: "500m"
```

- 仮想マシンのデフォルト割り当ては 2Gi です。
- Linux カーネルは、システムの起動時に約 100Mi のメモリーを使用します。
- QEMU プロセスでは約 30Mi のメモリーを使用します。
- **virtiofsd** プロセスでは約 10Mi のメモリーを使用します。
- **shim-v2** プロセスでは約 20Mi のメモリーを使用します。

ゲストで種類にかかわらず、ファイルシステム I/O を実行すると、ファイルバッファーがゲストカーネルに割り当てられます。ファイルバッファーは、**virtiofsd** プロセスだけでなく、ホスト上の QEMU プロセスでもマッピングされます。たとえば、ゲストでファイルバッファーキャッシュ 300Mi を使用すると、QEMU と **virtiofsd** の両方が、追加で 300Mi を使用するように見えます。ただし、3 つのケースすべてで同じメモリーが使用されています。つまり、メモリーの合計使用量は 300Mi のみで、このメモリー量が 3 つの異なる場所にマッピングされています。これは、メモリー使用量メトリクスの報告時に適切に考慮されます。

関連情報

- インストーラーでプロビジョニングされるクラスターのベアメタルへのデプロイ

3.2. WEB コンソールを使用した OPENSHIFT サンドボックスコンテナー OPERATOR のデプロイ

Web コンソールから Operator をインストールし、ワークロードを表示できます。

3.2.1. Web コンソールを使用した OpenShift サンドボックスコンテナー Operator のインストール

OpenShift Container Platform Web コンソールから OpenShift サンドボックスコンテナー Operator をインストールできます。

前提条件

- OpenShift Container Platform 4.8 がインストールされていること。
- **cluster-admin** ロールを持つユーザーとしてクラスターにアクセスできる。

手順

1. ブラウザーウィンドウを開き、OpenShift Container Platform Web コンソールにログインします。
2. **Administrator** パースペクティブで、**Operators → OperatorHub** に移動します。
3. **Filter by keyword** フィールドに **OpenShift sandboxed containers** と入力します。
4. **OpenShift sandboxed containers** タイルを選択します。
5. Operator についての情報を確認してから、**Install** をクリックします。
6. **Install Operator** ページで以下を行います。
   a. 利用可能な **Update Channel** オプションから **preview-1.0** を選択します。これにより、OpenShift Container Platform バージョンと互換性がある OpenShift サンドボックスコンテナーのバージョンをインストールすることができます。
   b. **インストールされた namespace** の場合、Operator recommended namespace オプションが選択されていることを確認します。これにより、Operator が必須の **openshift-sandboxed-containers-operator** namespace にインストールされます。この namespace は存在しない場合は、自動的に作成されます。

      注記

      OpenShift サンドボックスコンテナー Operator を **openshift-sandboxed-containers-operator** 以外の namespace にインストールしようとすると、インストールが失敗します。

   c. **Approval Strategy** の場合、デフォルト値である **Automatic** が選択されていることを確認します。OpenShift サンドボックス は、z-stream の新規リリースが利用可能になると自動的に更新されます。
7. **Install** をクリックし、Operator を OpenShift サンドボックスコンテナー namespace で利用可能にします。

これで、OpenShift サンドボックスコンテナー Operator がクラスターにインストールされました。クラスターでランタイムを有効にして Operator をトリガーすることができます。これには、OpenShift CLI (**oc**) を使用して **KataConfig** カスタムリソースを作成します。

```
apiVersion: kataconfiguration.openshift.io/v1
kind: KataConfig
metadata:
  name: example-kataconfig
```

3.2.2. Web コンソールからの OpenShift サンドボックスコンテナーワークロードの表示

OpenShift サンドボックスコンテナーベースのワークロードは、Web コンソールで表示すると通常のワークロードと同じように表示されます。この 2 つの唯一の違いは **runtimeClassName** のみです。**runtimeClassName** は、ワークロードに使用されるランタイムを決定します。このコンテキストでは、OpenShift がサンドボックスコンテナーベースで有効化されるランタイムは **kata** です。ワークロードの Pod が使用する **runtimeClass** を表示できます。

前提条件

- OpenShift Container Platform 4.8 をお使いのクラスターがインストールされている。
- **cluster-admin** ロールを持つユーザーとしてクラスターにアクセスできる。

手順

1. **Administration → Workloads** に移動します。
2. 詳細を表示するワークロードのタイプを特定します。たとえば、**Pod**、**Deployment**、**DeploymentConfigs** オブジェクトなどです。
3. 一覧から該当するワークロードを選択します。
4. **Details** ページで **runtimeClass** に移動します。
5. 詳細を表示するには、**runtimeClass** にカーソルを合わせます。**kata** がランタイムとして使用されている場合、**runtimeClass** の値は **kata** になります。

3.3. CLI を使用したサンドボックスコンテナー OPERATOR のインストール

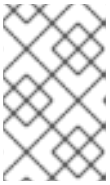
CLI から Operator をインストールし、ワークロードを表示できます。

3.3.1. CLI を使用したサンドボックスコンテナー Operator のインストール

OpenShift Container Platform CLI から OpenShift サンドボックスコンテナー Operator をインストールできます。

前提条件

- OpenShift Container Platform 4.8 をお使いのクラスターがインストールされている。
- OpenShift CLI (**oc**) がインストールされている。
- **cluster-admin** ロールを持つユーザーとしてクラスターにアクセスできる。
- OpenShift サンドボックスコンテナーカタログにサブスクライブしている。



注記

OpenShift サンドボックスコンテナーカタログにサブスクライブすると、**openshift-sandboxed-containers-operator** namespace の OpenShift サンドボックスコンテナー Operator にアクセスできるようになります。

手順

1. 以下のマニフェストを含む YAML ファイルを作成します。
   ■

```
apiVersion: v1
kind: Namespace
metadata:
  name: openshift-sandboxed-containers-operator
---
apiVersion: operators.coreos.com/v1
kind: OperatorGroup
metadata:
  name: openshift-sandboxed-containers-kataconfig-group
  namespace: openshift-sandboxed-containers-operator
spec:
  targetNamespaces:
    - openshift-sandboxed-containers-operator
---
apiVersion: operators.coreos.com/v1alpha1
kind: Subscription
metadata:
  name: sandboxed-containers-operatorhub
  namespace: openshift-sandboxed-containers-operator
spec:
  source: redhat-operators
  sourceNamespace: openshift-marketplace
  name: sandboxed-containers-operator
  startingCSV: sandboxed-containers-operator.v1.0.0
  channel: "preview-1.0"
  approval: "Automatic"
```

注記

preview-1.0 チャネルを使用することで、OpenShift Container Platform バージョンと互換性のある OpenShift Virtualization のバージョンをインストールできます。

2. OpenShift サンドボックスコンテナーに必要な **Namespace**、**OperatorGroup**、および **Subscription** オブジェクトを作成します。

```
$ oc create -f <file name>.yaml
```

3. Operator が正常にインストールされていることを確認します。

```
$ oc get csv -n openshift-sandboxed-containers-operator
```

出力例

```
NAME                         DISPLAY                          VERSION  REPLACES
PHASE
openshift-sandboxed-containers   openshift-sandboxed-containers-operator  1.0.0    <csv-of-
previous-version>   Succeeded
```

4. 利用可能なデプロイメントを表示します。

```
$ oc get deployments -n openshift-sandboxed-containers-operator
```

出力例

```
NAME                                READY  UP-TO-DATE   AVAILABLE   AGE
openshift-sandboxed-containers-operator                 1/111       9m48s
```

検証

- **KataConfig** リソースを作成してインストールをトリガーできるように Operator が稼働していることを確認します。

```
$ oc get deployments -n openshift-sandboxed-containers-operator
```

出力例

```
NAME                                    READY   UP-TO-DATE   AVAILABLE   AGE
openshift-sandboxed-containers-controller-manager   1/1     1            1           40d
```

関連情報

- CLI を使用した OperatorHub からのインストール

3.3.1.1. Kata ランタイムのインストールのトリガー

OpenShift サンドボックスコンテナー Operator をトリガーするために **KataConfig** カスタムリソース (CR) を 1 つ作成する必要があります。

- QEMU および **kata-containers** など、必要な RHCOS 拡張を RHCOS ノードにインストールします。
- ランタイム CRI-O が正しい Kata ランタイムハンドラーで設定されていることを確認します。
- 仮想化が原因となる追加のオーバーヘッドおよび必須の追加プロセスに必要な設定を使用して **RuntimeClass** カスタムリソースを作成します。

前提条件

- OpenShift Container Platform 4.8 をお使いのクラスターがインストールされている。
- OpenShift CLI (**oc**) がインストールされている。
- **cluster-admin** ロールを持つユーザーとしてクラスターにアクセスできる。

手順

1. **KataConfig** リソースを作成します。

```
$ oc create -f <file name>.yaml
```

例

```
apiVersion: kataconfiguration.openshift.io/v1
kind: KataConfig
metadata:
```

```
name: cluster-kataconfig
```

2. インストールの進捗を監視します。

- **KataConfig** インストールを記述できます。

```
$ oc describe kataconfig
```

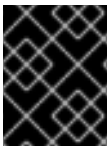
  - ステータスの **Completed nodes** フィールドを確認します。
  - **Completed nodes** の値がワーカーノードの数と一致する場合に、インストールは完了します。ステータスには、インストールが完了したノードの一覧も含まれます。

- **KataConfig** リソースを監視して、インストールの進捗を確認できます。

```
$ watch -n 10 oc describe kataconfig
```

  または、**KataConfig** リソースのステータスを確認することもできます。これには、**oc get KataConfig <name> -oyaml** を実行して、出力の **status** フィールドを検査します。

Kata ランタイムがクラスターにインストールされ、セカンダリーランタイムとして使用できるようになりました。クラスターの Kata 用に新たに作成された **RuntimeClass** が表示されることを確認します。



重要

OpenShift サンドボックスコンテナーは、Kata を主なランタイムとしてではなく、クラスターでセカンダリーオプションのランタイムとしてのみインストールします。

検証

- 以下を実行して **KataConfig** カスタムリソースの値をモニターできます。

```
$ watch oc describe KataConfig cluster-kataconfig
```

関連情報

- RuntimeClass

3.3.1.2. OpenShift サンドボックスコンテナーのノードの選択

特定のワーカーに Kata ランタイムを一部選択してインストールできます。

前提条件

- OpenShift Container Platform 4.8 をお使いのクラスターがインストールされている。
- OpenShift CLI (oc) がインストールされている。
- **cluster-admin** ロールを持つユーザーとしてクラスターにアクセスできる。

手順

1. ノードの選択に使用するラベルを特定します。この例では、ラベルを使用して OpenShift

1. ノードの選択に使用するラベルを特定します。この例では、ラベルを使用して、OpenShift サンドボックスコンテナーのワークロードで実行する候補として選択します。ノードが存在する場合は、それらが選択されます。

   a. ラベルをノードに適用するには、以下のコマンドを実行します。

   ```
   $ oc label node <worker_node_name> <label>=<value>
   ```

   これにより、ワーカーノードに値が **<value>** の **<label>** ラベルを付けます。

2. ラベルセレクターを追加するには、**KataConfig** カスタムリソース (CR) を編集します。

   ```
   $ oc edit kataconfig
   ```

   例

   ```
   apiVersion: kataconfiguration.openshift.io/v1
   kind: KataConfig
   metadata:
     name: cluster-kataconfig
   spec:
     kataConfigPoolSelector:
       matchLabels:
         custom-kata-machine-pool: 'true'
   ```

検証

- **machine-config-pool** オブジェクトのノードが設定の更新を行うかどうかを確認できます。
  - デフォルトノードを使用している場合は、以下を実行して **machine-config-pool** リソースをモニターできます。

    ```
    $ watch oc get mcp worker
    ```

  - 選択したノードを使用している場合は、以下を実行して **machine-config-pool** リソースをモニターできます。

    ```
    $ watch oc get mcp kata-oc
    ```

- **watch oc describe kataconfig cluster-kataconfig** を実行して、ノードで **sandboxed-containers** 拡張の失敗に関する情報を表示できます。この情報は、**machine-config-pool** オブジェクトのステータスから収集されます。以下を実行して情報を表示できます。

  ```
  $ oc describe mcp <machine-config-pool>
  ```

3.3.1.3. OpenShift サンドボックスコンテナーワークロードのスケジューリング

ワークロードをスケジューリングすると、OpenShift サンドボックスコンテナーで実行できます。

前提条件

- OpenShift Container Platform 4.8 をお使いのクラスターがインストールされている。
- OpenShift CLI (**oc**) がインストールされている。

- **cluster-admin** ロールを持つユーザーとしてクラスターにアクセスできる。

手順

1. **runtimeClassName: kata** を Pod でテンプレート化されたリソースに追加します。
   - **Pod** オブジェクト
   - **ReplicaSet** オブジェクト
   - **ReplicationController** オブジェクト
   - **StatefulSet** オブジェクト
   - **Deployment** オブジェクト
   - **DeploymentConfig** オブジェクト

Pod オブジェクトの例

```
apiVersion: v1
kind: Pod
metadata:
 name: mypod
spec:
  runtimeClassName: kata
```

Deployment オブジェクトの例

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: mypod
  labels:
    app: mypod
spec:
  replicas: 3
  selector:
    matchLabels:
      app: mypod
  template:
    metadata:
      labels:
        app: mypod
    spec:
      runtimeClassName: kata
      containers:
      - name: mypod
        image: myImage
```

Pod でテンプレート化されたリソースが **runtimeClassName: kata** によって作成されると、OpenShift Container Platform は OpenShift サンドボックスコンテナーでワークロードのスケジューリングを開始します。セレクターを使用しない場合には、デフォルト値がすべてのワーカーノードに設定されます。ワークロードが OpenShift サンドボックスコンテナーワークロードで実行されます。

3.3.2. CLI からの OpenShift サンドボックスコンテナーワークロードの表示

ワークロードの Pod が CLI から使用する **runtimeClass** を表示できます。

前提条件

- OpenShift Container Platform 4.8 をお使いのクラスターがインストールされている。
- OpenShift CLI (**oc**) がインストールされている。
- **cluster-admin** ロールを持つユーザーとしてクラスターにアクセスできる。

手順

- Pod の **runtimeClassName** フィールドを検査して、OpenShift サンドボックスコンテナーで実行している Pod と通常のコンテナーを比較します。
  - ノードでは、Pod ごとに対応する **qemu** プロセスがあります。

検証

- **openshift-sandboxed-containers-operator** コントローラー Pod のログをチェックして、実行されているステップに関する詳細なメッセージを確認できます。
  - コントローラー Pod 名を取得するには、以下を実行します。

    ```
    $ oc get pods -n openshift-sandboxed-containers-operator | grep openshift-sandboxed-containers-operator-controller-manager
    ```

    これにより、Pod のコンテナーマネージャーのログを監視できます。

第4章 OPENSHIFT サンドボックスコンテナーのアンインストール

4.1. WEB コンソールを使用した OPENSHIFT サンドボックスコンテナーのアンインストール

OpenShift Container Platform Web コンソールを使用して OpenShift サンドボックスコンテナーをアンインストールできます。

4.1.1. OpenShift サンドボックスコンテナーリソースの削除

OpenShift サンドボックスコンテナーをアンインストールするには、まず OpenShift サンドボックスコンテナーカスタムリソース **KataConfig** を削除する必要があります。これにより、**kata** ランタイムと関連リソースがクラスターから削除され、アンインストールされます。

前提条件

- OpenShift Container Platform 4.8 をお使いのクラスターがインストールされている。
- **cluster-admin** ロールを持つユーザーとしてクラスターにアクセスできる。
- **kata** を **runtimeClassName** として使用する実行中の Pod がない。
  - OpenShift CLI (**oc**) がインストールされている。
  - コマンドライン JSON プロセッサー (**jq**) がインストールされている。
  - 以下のコマンドを実行して、**kata** を **runtimeClassName** として使用する Pod が実行されていないことを確認します。

```
$ oc get pods -A -o json | jq -r '.items[] | select(.spec.runtimeClassName |
test("kata")).metadata.name'
```

手順

1. **kata** の値で **runtimeClassName** を使用するすべての Pod を削除します。
2. OpenShift Container Platform Web コンソールから、**Projects** 一覧より **openshift-sandboxed-containers** を選択します。
3. **Operators → Installed Operators** ページに移動します。
4. **OpenShift sandboxed containers** をクリックします。
5. **OpenShift sandboxed containers Operator** タブをクリックします。
6. **Operator Details** のスクロールダウン一覧をクリックし、**Delete KataConfig** をクリックします。
7. 確認ウィンドウで **Delete** をクリックします。

4.1.1.1. Web コンソールを使用した namespace の削除

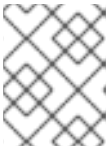
OpenShift Container Platform Web コンソールを使用して namespace を削除できます。

前提条件

- OpenShift Container Platform 4.8 をお使いのクラスターがインストールされている。
- **cluster-admin** ロールを持つユーザーとしてクラスターにアクセスできる。

手順

1. **Administration → Namespaces** に移動します。
2. namespace の一覧で削除する **openshift-sandboxed-containers-operator** namespace を見つけます。
3. namespace の一覧の右端で、**Options** メニュー から **Delete Namespace** を選択します。
4. **Delete Namespace** ペインが表示されたら、フィールドに **openshift-sandboxed-containers-operator** を入力します。



注記

Delete Namespace オプションが選択できない場合には、namespace を削除するパーミッションがありません。

5. **Delete** をクリックします。

4.1.2. OpenShift サンドボックスコンテナー Operator の削除

カタログサブスクリプションを削除し、Operator への namespace アクセスを取り消すことで、OpenShift でサンドボックス化したコンテナー Operator を削除できます。

前提条件

- OpenShift Container Platform 4.8 をお使いのクラスターがインストールされている。
- **cluster-admin** ロールを持つユーザーとしてクラスターにアクセスできる。

手順

1. **Operators → OperatorHub** ページに移動します。
2. **OpenShift sandboxed containers** 検索して、Operator を選択します。
3. **Uninstall** をクリックします。
4. **openshift-sandboxed-containers-operator** namespace を削除します。

4.2. CLI からの KATA ランタイムのアンインストール

OpenShift Container Platform コマンドラインインターフェイス (CLI) を使用して OpenShift サンドボックスコンテナーをアンインストールできます。

4.2.1. OpenShift サンドボックスコンテナーリソースの削除

kata ランタイムとその関連リソースすべて (CRI-O 設定や **RuntimeClass** など) をクラスターから削除およびアンインストールできます。

前提条件

- OpenShift Container Platform 4.8 をお使いのクラスターがインストールされている。
- OpenShift CLI (**oc**) がインストールされている。
- **cluster-admin** ロールを持つユーザーとしてクラスターにアクセスできる。

手順

1. 以下のコマンドを実行して **KataConfig** カスタムリソースを削除します。

```
$ oc delete kataconfig <KataConfig_CR_Name>
```

2. 以下のコマンドを実行して **KataConfig** カスタムリソース定義を削除します。

```
$ oc delete crd kataconfigs.kataconfiguration.openshift.io
```

OpenShift サンドボックスコンテナー Operator は、クラスターでランタイムを有効化するために初期に作成されていたリソースをすべて削除します。上記のコマンドを実行すると、インストールプロセス前の状態にクラスターが復元されます。**openshift-sandboxed-containers-operator** namespace が削除できるようになりました。

検証

- **KataConfig** カスタムリソースが削除されたことを確認するには、以下のコマンドを実行します。

```
$ oc get kataconfig <KataConfig_CR_Name>
```

出力例

```
No KataConfig instances exist
```

- **KataConfig** カスタムリソース定義が削除されていることを確認するには、以下のコマンドを実行します。

```
$ oc get crd kataconfigs.kataconfiguration.openshift.io
```

出力例

```
Unknown CR KataConfig
```

4.2.2. OpenShift サンドボックスコンテナー Operator の削除

OpenShift サンドボックスコンテナー Operator をクラスターから削除できます。

前提条件

- OpenShift Container Platform 4.8 をお使いのクラスターがインストールされている。
- OpenShift CLI (**oc**) がインストールされている。
- **cluster-admin** ロールを持つユーザーとしてクラスターにアクセスできる。

手順

1. 以下のコマンドを実行して、OpenShift サンドボックスコンテナー Operator サブスクリプションを Operator Lifecyle Manager (OLM) から削除します。

```
$ oc delete subscription openshift-sandboxed-containers-subscription -n openshift-sandboxed-containers-operator
```

2. 以下のコマンドを実行して、OpenShift サンドボックスコンテナーのクラスターサービスバージョン (CSV) 名を環境変数として設定します。

```
CSV_NAME=$(oc get csv -n openshift-sandboxed-containers-operator -o=custom-columns=:metadata.name)
```

3. 以下のコマンドを実行して、OpenShift サンドボックスコンテナーの CSV 名を削除します。

```
$ oc delete csv ${CSV_NAME} -n openshift-sandboxed-containers-operator
```

第5章 OPENSHIFT サンドボックスコンテナーのアップグレード

OpenShift サンドボックスコンテナー Operator および OpenShift サンドボックスコンテナーのアーティファクトをアップグレードして、OpenShift サンドボックスコンテナーのコンポーネントをアップグレードできます。

5.1. OPENSHIFT サンドボックスコンテナー OPERATOR のアップグレード

Operator Lifecycle Manager (OLM) を使用して、OpenShift サンドボックスコンテナー Operator を手動で設定するか、または自動的にアップグレードできます。最初のデプロイメント時に手動または自動アップグレードを選択できます。手動アップグレードのコンテキストでは、Web コンソールに、クラスター管理者がインストールでき、利用可能な更新を表示します。

追加リソース

- インストールされた Operator のアップグレード

5.2. OPENSHIFT サンドボックスコンテナーアーティファクトをアップグレードします。

OpenShift サンドボックスコンテナーアーティファクトは、Red Hat Enterprise Linux CoreOS (RHCOS) 拡張機能を使用してクラスターにデプロイされます。

RHCOS 拡張 **サンドボックスコンテナー** には、Kata コンテナーランタイム、ハイパーバイザーの QEMU およびその他の依存関係などの Kata コンテナーの実行に必要なコンポーネントが含まれます。この拡張機能は、クラスターを OpenShift Container Platform の新規リリースにアップグレードする時に、アップグレードされます。