



Red Hat OpenShift Container Storage 4.7

Amazon Web サービスを使用した OpenShift Container Storage のデプロイ

OpenShift Container Storage の OpenShift Container Platform AWS クラスターへの
インストールおよび設定方法

Red Hat OpenShift Container Storage 4.7 Amazon Web サービスを使用した OpenShift Container Storage のデプロイ

OpenShift Container Storage の OpenShift Container Platform AWS クラスターへのインストール
および設定方法

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

ローカルまたはクラウドストレージの Amazon Web Services を使用して Red Hat OpenShift Container Storage 4.7 をインストールする方法については、本書をお読みください。

目次

多様性を受け入れるオープンソースの強化	3
RED HAT ドキュメントへのフィードバック (英語のみ)	4
はじめに	5
第1章 RED HAT OPENSIFT CONTAINER STORAGE のデプロイの準備	6
1.1. RED HAT ENTERPRISE LINUX ベースのノード上のコンテナでのファイルシステムアクセスの有効化	6
1.2. VAULT でのキー値のバックエンドパスおよびポリシーの有効化	7
1.3. ローカルストレージデバイスを使用して OPENSIFT CONTAINER STORAGE をインストールするための要件	8
第2章 動的ストレージデバイスを使用したデプロイ	9
2.1. RED HAT OPENSIFT CONTAINER STORAGE OPERATOR のインストール	9
2.2. 内部モードでの OPENSIFT CONTAINER STORAGE CLUSTER SERVICE の作成	10
第3章 ローカルストレージデバイスを使用したデプロイ	14
3.1. 内部ローカルストレージを使用したデプロイの概要	14
3.2. RED HAT OPENSIFT CONTAINER STORAGE OPERATOR のインストール	14
3.3. ローカルストレージ OPERATOR のインストール	16
3.4. 利用可能なストレージデバイスの検索	16
3.5. AMAZON EC2 (ストレージ最適化: I3EN.2XLARGE インスタンスタイプ) での OPENSIFT CONTAINER STORAGE クラスターの作成	17
第4章 内部モードの OPENSIFT CONTAINER STORAGE デプロイメントの確認	22
4.1. POD の状態の確認	22
4.2. OPENSIFT CONTAINER STORAGE クラスターが正常であることの確認	23
4.3. MULTICLOUD OBJECT GATEWAY が正常であることの確認	24
4.4. OPENSIFT CONTAINER STORAGE 固有のストレージクラスが存在することの確認	25
第5章 OPENSIFT CONTAINER STORAGE のアンインストール	26
5.1. 内部モードでの OPENSIFT CONTAINER STORAGE のアンインストール	26
5.2. OPENSIFT CONTAINER STORAGE からのモニタリングスタックの削除	34
5.3. OPENSIFT CONTAINER STORAGE からの OPENSIFT CONTAINER PLATFORM レジストリーの削除	37
5.4. OPENSIFT CONTAINER STORAGE からのクラスターロギング OPERATOR の削除	38

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、[弊社の CTO である Chris Wright のメッセージ](#) を参照してください。

RED HAT ドキュメントへのフィードバック (英語のみ)

弊社のドキュメントについてのご意見をお聞かせください。ドキュメントの改善点があれば、ぜひお知らせください。フィードバックをお寄せいただくには、以下をご確認ください。

- 特定の部分についての簡単なコメントをお寄せいただく場合は、以下をご確認ください。
 1. ドキュメントの表示が **Multi-page HTML** 形式になっていることを確認してください。ドキュメントの右上隅に **Feedback** ボタンがあることを確認してください。
 2. マウスカーソルを使用して、コメントを追加するテキストの部分を強調表示します。
 3. 強調表示されたテキストの下に表示される **Add Feedback** ポップアップをクリックします。
 4. 表示される指示に従ってください。
- より詳細なフィードバックをお寄せいただく場合は、Bugzilla のチケットを作成してください。
 1. [Bugzilla](#) の Web サイトに移動します。
 2. Component (コンポーネント) として **Documentation** を使用します。
 3. **Description** フィールドに、ドキュメントの改善に向けたご提案を記入してください。ドキュメントの該当部分へのリンクも追加してください。
 4. **Submit Bug** をクリックします。

はじめに

Red Hat OpenShift Container Storage 4.7 は、接続環境または非接続環境での既存の Red Hat OpenShift Container Platform (RHOC) AWS クラスターへのデプロイメントをサポートし、プロキシ環境に対する追加設定なしのサポートを提供します。



注記

AWS では、内部の Openshift Container Storage クラスターのみがサポートされます。デプロイメントについての詳細は、[デプロイメントのプランニング](#) および [OpenShift Container Storage のデプロイの準備](#) について参照してください。

OpenShift Container Storage をデプロイするには、[OpenShift Container Storage のデプロイの準備](#) についての章の要件を確認し、環境についての以下のデプロイメントプロセスのいずれかを実行します。

- [動的ストレージデバイスを使用したデプロイ](#)
- [ローカルストレージデバイスを使用したデプロイ](#) [テクノロジープレビュー]

第1章 RED HAT OPENSIFT CONTAINER STORAGE のデプロイの準備

動的またはローカルストレージデバイスを使用して OpenShift Container Storage を OpenShift Container Platform にデプロイすると、内部クラスターリソースを作成するオプションが提供されます。これにより、ベースサービスの内部プロビジョニングが可能になり、追加のストレージクラスをアプリケーションで使用可能にすることができます。

OpenShift Container Storage のデプロイを開始する前に、以下を実行します。

1. ワーカーノード向けの Red Hat Enterprise Linux ベースのホストについては、[Red Hat Enterprise Linux ベースのノードでのコンテナのファイルシステムのアクセスを有効にします](#)。



注記

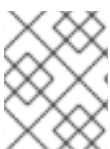
Red Hat Enterprise Linux CoreOS(RHCOS) の場合は、この手順を省略します。

2. オプション: 外部鍵管理システム (KMS) を使用してクラスター全体の暗号化を有効にする場合:
 - トークンのあるポリシーが存在し、Vault のキー値のバックエンドパスが有効にされていることを確認します。[enabled the key value backend path and policy in Vault](#) を参照してください。
 - Vault サーバーで署名済みの証明書を使用していることを確認します。
3. ノードの最小要件 [テクノロジープレビュー]

OpenShift Container Storage クラスターは、標準のデプロイメントリソース要件を満たしていない場合に、最小の設定でデプロイされます。プランニングガイドの [リソース要件](#) セクションを参照してください。
4. [ローカルストレージデバイスを使用して OpenShift Container Storage をインストールするための要件](#)を確認します。これは、動的ストレージデバイスを使用するデプロイメントには該当しません。

1.1. RED HAT ENTERPRISE LINUX ベースのノード上のコンテナでのファイルシステムアクセスの有効化

ユーザーによってプロビジョニングされるインフラストラクチャー (UPI) で Red Hat Enterprise Linux がベースの OpenShift Data Foundation にワーカーノードを含めて OpenShift Container Storage をデプロイしても自動的に、基盤の Ceph ファイルシステムへのコンテナアクセスが提供されるわけではありません。



注記

Red Hat Enterprise Linux CoreOS(RHCOS) をベースとするホストの場合は、この手順を省略します。

手順

1. Red Hat Enterprise Linux ベースのノードにログインし、ターミナルを開きます。
2. クラスター内の各ノードについて、以下を実行します。

- a. ノードが `rhel-7-server-extras-rpms` リポジトリにアクセスできることを確認します。

```
# subscription-manager repos --list-enabled | grep rhel-7-server
```

出力に **rhel-7-server-rpms** と **rhel-7-server-extras-rpms** の両方が表示されない場合は、以下のコマンドを実行して各リポジトリを有効にします。

```
# subscription-manager repos --enable=rhel-7-server-rpms
# subscription-manager repos --enable=rhel-7-server-extras-rpms
```

- b. 必要なパッケージをインストールします。

```
# yum install -y policycoreutils container-selinux
```

- c. SELinux での Ceph ファイルシステムのコンテナの使用を永続的に有効にします。

```
# setsebool -P container_use_cephfs on
```

1.2. VAULT でのキー値のバックエンドパスおよびポリシーの有効化

前提条件

- Vault への管理者アクセス。
- 注: 後に変更することはできないため、命名規則に基づいてバックエンド **path** として一意のパス名を選択します。

手順

1. Vault で Key/Value (KV) バックエンドパスを有効にします。
Vault KV シークレットエンジン API の場合は、バージョン 1 を使用します。

```
$ vault secrets enable -path=ocs kv
```

Vault KV シークレットエンジン API の場合は、バージョン 2 です。

```
$ vault secrets enable -path=ocs kv-v2
```

2. 以下のコマンドを使用して、シークレットでの書き込み操作または削除操作の実行をユーザーを制限するポリシーを作成します。

```
echo '
path "ocs/*" {
  capabilities = ["create", "read", "update", "delete", "list"]
}
path "sys/mounts" {
  capabilities = ["read"]
}' | vault policy write ocs -
```

3. 上記のポリシーに一致するトークンを作成します。

```
$ vault token create -policy=ocs -format json
```

1.3. ローカルストレージデバイスを使用して OPENSHIFT CONTAINER STORAGE をインストールするための要件

ノードの要件

クラスターは、それぞれローカルに接続されたストレージデバイスを持つ 3 つ以上の OpenShift Container Platform ワーカーノードで設定される必要があります。

- 選択した 3 つのノードには、OpenShift Container Storage で使用できる raw ブロックデバイスが少なくとも 1 つ必要です。
- 使用するデバイスは空である必要があります。ディスクには物理ボリューム (PV)、ボリュームグループ (VG)、または論理ボリューム (LV) を含めないでください。
- 3 つ以上のラベルが付けられたノードが必要です。
 - ノードが複数のアベイラビリティゾーンプラットフォームの異なる場所/アベイラビリティゾーンに分散されていることを確認します。
 - OpenShift Container Storage によって使用されるローカルストレージデバイスを持つ各ノードには、OpenShift Container Storage Pod をデプロイするための特定のラベルが必要です。ノードにラベルを付けるには、以下のコマンドを使用します。

```
$ oc label nodes <NodeNames> cluster.ocs.openshift.io/openshift-storage=
```

プランニングガイドの [リソース要件](#) のセクションを参照してください。

ノードの最小要件 [テクノロジープレビュー]

OpenShift Container Storage クラスターは、標準のデプロイメントリソース要件を満たしていない場合に、最小の設定でデプロイされます。プランニングガイドの [リソース要件](#) のセクションを参照してください。

第2章 動的ストレージデバイスを使用したデプロイ

AWS EBS (タイプ: gp2) で提供される動的ストレージデバイスを使用して OpenShift Container Storage を OpenShift Container Platform にデプロイすると、内部クラスターリソースを作成するオプションが提供されます。これにより、ベースサービスの内部プロビジョニングが可能になり、追加のストレージクラスをアプリケーションで使用可能にすることができます。



注記

AWS では、内部の OpenShift Container Storage クラスターのみがサポートされます。デプロイメントの要件についての詳細は、[Planning your deployment](#) を参照してください。

[OpenShift Container Storage のデプロイの準備](#) についての章にある要件に対応していることを確認してから、動的ストレージデバイスを使用したデプロイについて以下の手順を実行してください。

1. [Red Hat OpenShift Container Storage Operator をインストールする](#)
2. [OpenShift Container Storage Cluster Service を作成します](#)。

2.1. RED HAT OPENSIFT CONTAINER STORAGE OPERATOR のインストール

Red Hat OpenShift Container Storage は、Red Hat OpenShift Container Platform Operator Hub を使用してインストールできます。

前提条件

- cluster-admin および Operator インストールのパーミッションを持つアカウントを使用して OpenShift Container Platform クラスターにアクセスできること。
- RHOCV クラスターにワーカーノードが少なくとも 3 つある。
- その他のリソース要件については、[デプロイメントのプランニング](#) を参照してください。



注記

- OpenShift Container Storage のクラスター全体でのデフォルトノードセクターを上書きする必要がある場合は、コマンドラインインターフェイスで以下のコマンドを使用し、**openshift-storage** namespace の空のノードセクターを指定できます (この場合、openshift-storage namespace を作成します)。

```
$ oc annotate namespace openshift-storage openshift.io/node-selector=
```

- ノードに Red Hat OpenShift Container Storage リソースのみがスケジュールされるように、そのノードに **infra** のテイントを設定します。これにより、サブスクリプションコストを節約できます。詳細は、ストレージリソースの管理および割り当てガイドの [Red Hat OpenShift Container Storage に専用のワーカーノードを使用する方法](#) の章を参照してください。

手順

1. Web コンソールで **Operators → OperatorHub** ページに移動し、クリックします。

2. スクロールするか、またはキーワードを Filter by keyword ボックスに入力し、OpenShift Container Storage Operator を検索します。
3. OpenShift Container Storage Operator ページで、**Install** をクリックします。
4. **Install Operator** ページで、以下の必須オプションがデフォルトで選択されます。
 - a. Channel を **stable-4.7** として更新します。
 - b. Installation Mode オプションに **A specific namespace on the cluster** を選択します。
 - c. Installed Namespace に **Operator recommended namespace openshift-storage** を選択します。namespace **openshift-storage** が存在しない場合、これは Operator のインストール時に作成されます。
 - d. **承認ストラテジー** を **Automatic** または **Manual** として選択します。
 - e. **Install** をクリックします。

Automatic (自動) 更新を選択している場合、Operator Lifecycle Manager (OLM) は介入なしに、Operator の実行中のインスタンスを自動的にアップグレードします。

Manual (手動) 更新を選択している場合、OLM は更新要求を作成します。クラスター管理者は、Operator が新規バージョンに更新されるように更新要求を手動で承認する必要があります。

検証手順

OpenShift Container Storage Operator に、インストールが正常に実行されたことを示す緑色のチェックマークが表示されていることを確認します。

次のステップ

- OpenShift Container Storage クラスターを作成します。
詳細は、[内部モードでの OpenShift Container Storage Cluster Service の作成](#) について参照してください。

2.2. 内部モードでの OPENSIFT CONTAINER STORAGE CLUSTER SERVICE の作成

以下の手順を使用して、OpenShift Container Storage Operator のインストール後に OpenShift Container Storage Cluster Service を作成します。

前提条件

- OpenShift Container Storage は Operator Hub からインストールする必要があります。詳細は、[Operator Hub を使用した OpenShift Container Storage Operator のインストール](#) について参照してください。

手順

1. OpenShift Web コンソールにログインします。
2. **Operators → Installed Operators** をクリックし、インストールされた Operator をすべて表示します。
選択された **Project** が **openshift-storage** であることを確認します。

3. Storage Cluster の **OpenShift Container Storage > Create Instance** リンクをクリックします。
4. **Mode** がデフォルトで **Internal** に設定されます。
5. **Select capacity and nodes** で、以下を実行します。
 - a. **Storage Class** を選択します。デフォルトでは **gp2** に設定されます。
 - b. ドロップダウンリストから **Requested Capacity** を選択します。デフォルトで、これは **2 TiB** に設定されます。ドロップダウンを使用して容量の値を変更できます。



注記

初期ストレージ容量を選択すると、クラスターの拡張は、選択された使用可能な容量を使用してのみ実行されます (raw ストレージの 3 倍)。

- c. **Select Nodes** セクションで、少なくとも 3 つの利用可能なノードを選択します。複数のアベイラビリティゾーンを持つクラウドプラットフォームの場合は、ノードが異なる場所/アベイラビリティゾーンに分散されていることを確認します。

選択したノードが集約された 30 CPU および 72 GiB の RAM の OpenShift Container Storage クラスターの要件と一致しない場合は、最小クラスターがデプロイされます。ノードの最小要件については、プランニングガイドの [リソース要件](#) セクションを参照してください。

- d. **Next** をクリックします。
6. (オプション) セキュリティ設定
 - a. **Enable encryption** チェックボックスを選択して、ブロックおよびファイルストレージを暗号化します。
 - b. 1 つまたは両方の **Encryption level** を選択します。
 - **クラスター全体の暗号化** クラスター全体 (ブロックおよびファイル) を暗号化します。
 - **Storage class encryption** (ストレージクラスの暗号化): 暗号化対応のストレージクラスを使用して暗号化された永続ボリューム (ブロックのみ) を作成します。



重要

ストレージクラスの暗号化は、RBD PV でのみ利用可能な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機能は Red Hat の実稼働環境でのサービスレベルアグリーメント (SLA) ではサポートされていないため、Red Hat では実稼働環境での使用を推奨していません。Red Hat は実稼働環境でこれらを使用することを推奨していません。これらの機能は、近々発表予定の製品機能をリリースに先駆けてご提供することにより、お客様は機能性をテストし、開発プロセス中にフィードバックをお寄せいただくことができます。

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。

- c. **Connect to an external key management service** チェックボックスを選択します。これはクラスター全体の暗号化の場合はオプションになります。

- i. **Key Management Service Provider**はデフォルトで **Vault** に設定されます。
 - ii. Vault **Service Name**、Vault サーバーのホスト **Address**('https://<hostname または ip>')、**Port number** および **Token** を入力します。
 - iii. **Advanced Settings** を展開して、Vault 設定に基づいて追加の設定および証明書の詳細を入力します。
 - A. OpenShift Container Storage 専用かつ特有のキー値のシークレットパスを **Backend Path** に入力します。
 - B. **TLS Server Name** および **Vault Enterprise Namespace** を入力します。
 - C. それぞれの PEM でエンコードされた証明書ファイルをアップロードして、**CA Certificate**、**Client Certificate** および **Client Private Key** を指定します。
 - D. **Save** をクリックします。
 - d. **Next** をクリックします。
7. 設定の詳細を確認します。設定を変更するには、**Back** をクリックして以前の設定ページに戻ります。
8. **Create** をクリックします。
9. Vault Key/Value (KV) シークレットエンジン API の場合に configmap を編集します。バージョン 2 は、鍵管理システム (KMS) のクラスター全体の暗号化に使用されます。
 - a. OpenShift Web コンソールで **Workloads → ConfigMaps** に移動します。
 - b. KMS 接続の詳細を表示するには、**ocs -kms-connection-details** をクリックします。
 - c. configmap を編集します。
 - i. **Action menu(⋮)→ Edit ConfigMap** をクリックします。
 - ii. **VAULT_BACKEND** パラメーターを **v2** に設定します。

```
kind: ConfigMap
apiVersion: v1
metadata:
  name: ocs-kms-connection-details
[...]
data:
  KMS_PROVIDER: vault
  KMS_SERVICE_NAME: vault
[...]
  VAULT_BACKEND: v2
[...]
```

- iii. **Save** をクリックします。

検証手順

1. ストレージクラスターの詳細ページで、ストレージクラスター名の横に緑色のチェックマークが表示され、クラスターが正常に作成されたことを示します。

2. インストールされたストレージクラスターの最後の **Status** が緑色のチェックマークと共に **Phase: Ready** と表示されていることを確認します。
 - **Operators → Installed Operators → Storage Cluster**のリンクをクリックして、ストレージクラスターのインストールのステータスを表示します。
 - または、Operator **Details** タブで、**Storage Cluster** タブをクリックすると、ステータスを表示できます。
3. OpenShift Container Storage のすべてのコンポーネントが正常にインストールされていることを確認するには、[OpenShift Container Storage インストールの確認](#) を参照してください。

第3章 ローカルストレージデバイスを使用したデプロイ

ローカルストレージデバイスを使用して OpenShift Container Storage を OpenShift Container Platform にデプロイすると、内部クラスターリソースを作成するオプションが提供されます。これにより、ベースサービスの内部プロビジョニングが可能になり、追加のストレージクラスをアプリケーションで使用可能にすることができます。

このセクションを使用して、OpenShift Container Platform がすでにインストールされている Amazon EC2 ストレージ最適化 I3 に OpenShift Container Storage をインストールします。

重要

ローカルストレージ Operator を使用した Amazon EC2 ストレージ最適化 I3 インスタンスへの OpenShift Container Storage のインストール機能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機能は Red Hat の実稼働環境でのサービスレベルアグリーメント (SLA) ではサポートされていないため、Red Hat では実稼働環境での使用を推奨していません。Red Hat は実稼働環境でこれらを使用することを推奨していません。これらの機能は、近々発表予定の製品機能をリリースに先駆けてご提供することにより、お客様は機能性をテストし、開発プロセス中にフィードバックをお寄せいただくことができます。Red Hat OpenShift Container Storage デプロイメントでは、3つのワーカーノードで実行されるアプリケーションやその他のワークロードを使用せずに、新規クラスターを使用することが想定されます。アプリケーションは追加のワーカーノードで実行されます。

また、[OpenShift Container Storage のデプロイの準備](#) についての章にある要件に対応していることを確認してから、以下の手順に進んでください。

3.1. 内部ローカルストレージを使用したデプロイの概要

ローカルストレージデバイスを使用した OpenShift Container Storage をデプロイするには、以下を実行します。

1. [Red Hat OpenShift Container Storage Operator をインストールします。](#)
2. [ローカルストレージ Operator のインストール](#)
3. [利用可能なストレージデバイスを見つけます。](#)
4. [Amazon EC2 \(ストレージ最適化: i3en.2xlarge インスタンスタイプ\) で OpenShift Container Storage クラスターを作成します。](#)

3.2. RED HAT OPENSIFT CONTAINER STORAGE OPERATOR のインストール

Red Hat OpenShift Container Storage は、Red Hat OpenShift Container Platform Operator Hub を使用してインストールできます。

前提条件

- cluster-admin および Operator インストールのパーミッションを持つアカウントを使用して OpenShift Container Platform クラスターにアクセスできること。
- RHOCV クラスターにワーカーノードが少なくとも3つある。

- その他のリソース要件については、[デプロイメントのプランニング](#)を参照してください。

注記

- OpenShift Container Storage のクラスター全体でのデフォルトノードセクターを上書きする必要がある場合は、コマンドラインインターフェイスで以下のコマンドを使用し、**openshift-storage** namespace の空のノードセクターを指定できます (この場合、openshift-storage namespace を作成します)。

```
$ oc annotate namespace openshift-storage openshift.io/node-selector=
```

- ノードに Red Hat OpenShift Container Storage リソースのみがスケジュールされるように、そのノードに **infra** のテイントを設定します。これにより、サブスクリプションコストを節約できます。詳細は、ストレージリソースの管理および割り当てガイドの [Red Hat OpenShift Container Storage に専用のワーカーノードを使用する方法](#) の章を参照してください。

手順

1. Web コンソールで **Operators → OperatorHub** ページに移動し、クリックします。
 2. スクロールするか、またはキーワードを Filter by keyword ボックスに入力し、OpenShift Container Storage Operator を検索します。
 3. OpenShift Container Storage Operator ページで、**Install** をクリックします。
 4. **Install Operator** ページで、以下の必須オプションがデフォルトで選択されます。
 - a. Channel を **stable-4.7** として更新します。
 - b. Installation Mode オプションに **A specific namespace on the cluster** を選択します。
 - c. Installed Namespace に **Operator recommended namespace openshift-storage** を選択します。namespace **openshift-storage** が存在しない場合、これは Operator のインストール時に作成されます。
 - d. **承認ストラテジー** を **Automatic** または **Manual** として選択します。
 - e. **Install** をクリックします。
- Automatic** (自動) 更新を選択している場合、Operator Lifecycle Manager (OLM) は介入なしに、Operator の実行中のインスタンスを自動的にアップグレードします。

Manual (手動) 更新を選択している場合、OLM は更新要求を作成します。クラスター管理者は、Operator が新規バージョンに更新されるように更新要求を手動で承認する必要があります。

検証手順

OpenShift Container Storage Operator に、インストールが正常に実行されたことを示す緑色のチェックマークが表示されていることを確認します。

次のステップ

- OpenShift Container Storage クラスターを作成します。
詳細は、[Amazon EC2 \(ストレージ最適化: i3en.2xlarge インスタンスタイプ\) での OpenShift Container Storage クラスターの作成](#) について参照してください。

3.3. ローカルストレージ OPERATOR のインストール

手順

1. OpenShift Web コンソールにログインします。
2. **Operators → OperatorHub** をクリックします。
3. **Filter by keyword...** ボックスに **local storage** と入力して、Operator のリストから **Local Storage** Operator を検索し、クリックします。
4. **Install** をクリックします。
5. **Install Operator** ページで、以下のオプションを設定します。
 - a. Channel を **stable-4.7** として更新します。
 - b. Installation Mode オプションに **A specific namespace on the cluster** を選択します。
 - c. Installed Namespace に **Operator recommended namespace openshift-local-storage** を選択します。
 - d. Approval Strategy に **Automatic** を選択します。
6. **Install** をクリックします。
7. ローカルストレージ Operator がステータス **Succeeded** を表示していることを確認します。

3.4. 利用可能なストレージデバイスの検索

以下の手順を使用して、PV を作成する前に、OpenShift Container Storage ラベル **cluster.ocs.openshift.io/openshift-storage=** でラベルを付けた 3 つ以上のノードのそれぞれのデバイス名を特定します。

手順

1. OpenShift Container Storage ラベルの付いたノードの名前の一覧を表示し、確認します。

```
$ oc get nodes -l cluster.ocs.openshift.io/openshift-storage=
```

出力例:

NAME	STATUS	ROLES	AGE	VERSION
ip-10-0-135-71.us-east-2.compute.internal	Ready	worker	6h45m	v1.16.2
ip-10-0-145-125.us-east-2.compute.internal	Ready	worker	6h45m	v1.16.2
ip-10-0-160-91.us-east-2.compute.internal	Ready	worker	6h45m	v1.16.2

2. OpenShift Container Storage リソースに使用される各ノードにログインし、利用可能な各 raw ブロックデバイスの一意の **by-id** デバイス名を見つけます。

```
$ oc debug node/<node name>
```

出力例:

```
$ oc debug node/ip-10-0-135-71.us-east-2.compute.internal
Starting pod/ip-10-0-135-71us-east-2computeinternal-debug ...
To use host binaries, run `chroot /host`
Pod IP: 10.0.135.71
If you don't see a command prompt, try pressing enter.
sh-4.2# chroot /host
sh-4.4# lsblk
NAME                                MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
xvda                                202:0    0 120G  0 disk
|-xvda1                            202:1    0  384M  0 part /boot
|-xvda2                            202:2    0 127M  0 part /boot/efi
|-xvda3                            202:3    0   1M  0 part
`-xvda4                            202:4    0 119.5G  0 part
   `--coreos-luks-root-nocrypt 253:0    0 119.5G  0 dm  /sysroot
nvme0n1                            259:0    0  2.3T  0 disk
nvme1n1                            259:1    0  2.3T  0 disk
```

この例では、利用可能なローカルデバイスは **nvme0n1** および **nvme1n1** です。

- 手順 2 で選択した各デバイスの一意の ID を特定します。

```
sh-4.4# ls -l /dev/disk/by-id/ | grep Storage
lrwxrwxrwx. 1 root root 13 Mar 17 16:24 nvme-
Amazon_EC2_NVMe_Instance_Storage_AWS10382E5D7441494EC -> ../../nvme0n1
lrwxrwxrwx. 1 root root 13 Mar 17 16:24 nvme-
Amazon_EC2_NVMe_Instance_Storage_AWS60382E5D7441494EC -> ../../nvme1n1
```

上記の例では、2 つのローカルデバイスの ID は以下になります。

- nvme0n1: nvme-Amazon_EC2_NVMe_Instance_Storage_AWS10382E5D7441494EC
 - nvme1n1: nvme-Amazon_EC2_NVMe_Instance_Storage_AWS60382E5D7441494EC
- 上記の手順を繰り返し、OpenShift Container Storage で使用されるストレージデバイスを持つその他のすべてのノードのデバイス ID を特定します。詳細は、[ナレッジベースアールクル](#) を参照してください。

3.5. AMAZON EC2 (ストレージ最適化: I3EN.2XLARGE インスタンスタイプ) での OPENSIFT CONTAINER STORAGE クラスターの作成

以下の手順を使用して、Amazon EC2(ストレージ最適化: i3en.2xlarge インスタンスタイプ) インフラストラクチャーに OpenShift Container Storage クラスターを作成します。以下が実行されます。

1. **LocalVolume** CR を使用して PV を作成する
2. 新しい **StorageClass** を作成する

Amazon EC2 (ストレージ最適化: i3en.2xlarge インスタンスタイプ) には、2 つの NVMe (non-volatile memory express) ディスクが含まれます。この手順の例では、このインスタンスタイプと共に提供される 2 つのディスクの使用方法について説明します。

AmazonEC2 I3 の一時ストレージを使用する場合は、以下を行います。

- 3 つのアベイラビリティゾーンを使用し、すべてのデータを失うリスクを軽減する。

- `ec2:StopInstances` パーミッションを持つユーザーの数を制限し、インスタンスを誤ってシャットダウンすることを回避する。



警告

OpenShift Container Storage の永続ストレージに Amazon EC2 I3 の一時ストレージを使用することは推奨されません。3 つのノードをすべて停止するとデータ損失が発生する可能性があるためです。

Amazon EC2 I3 の一時ストレージは、以下のようなシナリオでのみ使用することが推奨されます。

- 特定のデータ処理 (data crunching) のためにデータがある場所からコピーされるクラウドバースト (時間に制限がある) が想定される場合。
- 開発環境またはテスト環境が想定される場合。



重要

ローカルストレージ Operator を使用した Amazon EC2 ストレージ最適化 `i3en.2xlarge` インスタンスの OpenShift Container Storage のインストール機能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機能は Red Hat の実稼働環境でのサービスレベルアグリーメント (SLA) ではサポートされていないため、Red Hat では実稼働環境での使用を推奨していません。Red Hat は実稼働環境でこれらを使用することを推奨していません。これらの機能は、近々発表予定の製品機能をリリースに先駆けてご提供することにより、お客様は機能性をテストし、開発プロセス中にフィードバックをお寄せいただくことができます。

前提条件

- [ローカルストレージデバイスを使用した OpenShift Container Storage のインストールの要件](#)に記載されるすべての要件を満たしていることを確認します。
- OpenShift Container Platform ワーカーノードに OpenShift Container Storage ラベルを付けられていることを確認します。これは **nodeSelector** として使用されます。

```
oc get nodes -l cluster.ocs.openshift.io/openshift-storage -o jsonpath='{range .items[*]}{.metadata.name}{"\n"}'
```

出力例:

```
ip-10-0-135-71.us-east-2.compute.internal
ip-10-0-145-125.us-east-2.compute.internal
ip-10-0-160-91.us-east-2.compute.internal
```

手順

1. **LocalVolume** カスタムリソース (CR) を使用してストレージノードにローカル永続ボリューム (PV) を作成します。

OpenShift Storage Container ラベルをノードセクターおよび **by-id** デバイス識別子として使用する **LocalVolume** CR **local-storage-block.yaml** の例

```
apiVersion: local.storage.openshift.io/v1
kind: LocalVolume
metadata:
  name: local-block
  namespace: openshift-local-storage
  labels:
    app: ocs-storagecluster
spec:
  tolerations:
    - key: "node.ocs.openshift.io/storage"
      value: "true"
      effect: NoSchedule
  nodeSelector:
    nodeSelectorTerms:
      - matchExpressions:
          - key: cluster.ocs.openshift.io/openshift-storage
            operator: In
            values:
              - ""
  storageClassDevices:
    - storageClassName: localblock
      volumeMode: Block
      devicePaths:
        - /dev/disk/by-id/nvme-
        Amazon_EC2_NVMe_Instance_Storage_AWS10382E5D7441494EC # <-- modify this line
        - /dev/disk/by-id/nvme-
        Amazon_EC2_NVMe_Instance_Storage_AWS1F45C01D7E84FE3E9 # <-- modify this line
        - /dev/disk/by-id/nvme-
        Amazon_EC2_NVMe_Instance_Storage_AWS136BC945B4ECB9AE4 # <-- modify this line
        - /dev/disk/by-id/nvme-
        Amazon_EC2_NVMe_Instance_Storage_AWS10382E5D7441464EP # <-- modify this line
        - /dev/disk/by-id/nvme-
        Amazon_EC2_NVMe_Instance_Storage_AWS1F45C01D7E84F43E7 # <-- modify this line
        - /dev/disk/by-id/nvme-
        Amazon_EC2_NVMe_Instance_Storage_AWS136BC945B4ECB9AE8 # <-- modify this line
```

各 Amazon EC2 I3 インスタンスには 2 つのディスクがあり、この例ではそれぞれのノードで両方のディスクを使用します。

2. **LocalVolume** CR を作成します。

```
$ oc create -f local-storage-block.yaml
```

出力例:

```
localvolume.local.storage.openshift.io/local-block created
```

3. Pod が作成されているかどうかを確認します。

```
$ oc -n openshift-local-storage get pods
```

4. PV が作成されているかどうかを確認します。

3つのワーカーノード上の各ローカルストレージデバイスの新規 PV が表示される必要があります。ワークノードごとに利用可能な2つのストレージデバイス (各ノードに 2.3 TiB のサイズが設定される) について説明している、[利用可能なストレージデバイスの検索](#) についてのセクションの例を参照してください。

```
$ oc get pv
```

出力例:

NAME	CAPACITY	ACCESS MODES	RECLAIM POLICY	STATUS	CLAIM
STORAGECLASS	REASON	AGE			
local-pv-1a46bc79	2328Gi	RWO	Delete	Available	localblock 14m
local-pv-429d90ee	2328Gi	RWO	Delete	Available	localblock 14m
local-pv-4d0a62e3	2328Gi	RWO	Delete	Available	localblock 14m
local-pv-55c05d76	2328Gi	RWO	Delete	Available	localblock 14m
local-pv-5c7b0990	2328Gi	RWO	Delete	Available	localblock 14m
local-pv-a6b283b	2328Gi	RWO	Delete	Available	localblock 14m

5. **LocalVolume** CR の作成時に表示される新規の **StorageClass** の有無を確認します。この **StorageClass** は、以下の手順で **StorageCluster** PVC を提供するために使用されます。

```
$ oc get sc | grep localblock
```

出力例:

NAME	PROVISIONER	RECLAIMPOLICY
VOLUME	BINDINGMODE	ALLOWVOLUMEEXPANSION
AGE		
localblock	kubernetes.io/no-provisioner	Delete
WaitForFirstConsumer	false	15m

6. ローカルストレージ Operator で作成される PV を消費するために **localblock** StorageClass を使用する **StorageCluster** CR を作成します。

monDataDirHostPath および **localblock** StorageClass を使用する **StorageCluster** CR **ocs-cluster-service.yaml** の例。

```
apiVersion: ocs.openshift.io/v1
kind: StorageCluster
metadata:
  name: ocs-storagecluster
  namespace: openshift-storage
spec:
  manageNodes: false
  resources:
    mds:
      limits:
        cpu: 3
        memory: 8Gi
      requests:
        cpu: 1
        memory: 8Gi
  monDataDirHostPath: /var/lib/rook
  storageDeviceSets:
```



```
- count: 2
  dataPVCTemplate:
    spec:
      accessModes:
        - ReadWriteOnce
      resources:
        requests:
          storage: 2328Gi
      storageClassName: localblock
      volumeMode: Block
    name: ocs-deviceset
  placement: {}
  portable: false
  replica: 3
  resources:
    limits:
      cpu: 2
      memory: 5Gi
    requests:
      cpu: 1
      memory: 5Gi
```



重要

OSD でノード全体に Guaranteed サイズが保証されるようにするには、**storageDeviceSets** のストレージサイズを、ノードで作成される PV のサイズ以下に指定する必要があります。

7. **StorageCluster** CR を作成します。

```
$ oc create -f ocs-cluster-service.yaml
```

出力例

```
storagecluster.ocs.openshift.io/ocs-cluster-service created
```

検証手順

[OpenShift Container Storage インストールの検証](#) を参照してください。

第4章 内部モードの OPENSIFT CONTAINER STORAGE デプロイメントの確認

このセクションを使用して、OpenShift Container Storage が正常にデプロイされていることを確認します。

4.1. POD の状態の確認

OpenShift Container Storage が正常にデプロイされているかどうかを判別するために、Pod の状態が **Running** であることを確認できます。

手順

1. OpenShift Web コンソールの左側のペインから **Workloads** → **Pods** をクリックします。
2. **Project** ドロップダウンリストから **openshift-storage** を選択します。
各コンポーネントについて予想される Pod 数や、これがノード数によってどのように異なるかについての詳細は、[表4.1「OpenShift Container Storage クラスターに対応する Pod」](#) を参照してください。
3. **Running** および **Completed** タブをクリックして、以下の Pod が実行中および完了状態にあることを確認します。

表4.1 OpenShift Container Storage クラスターに対応する Pod

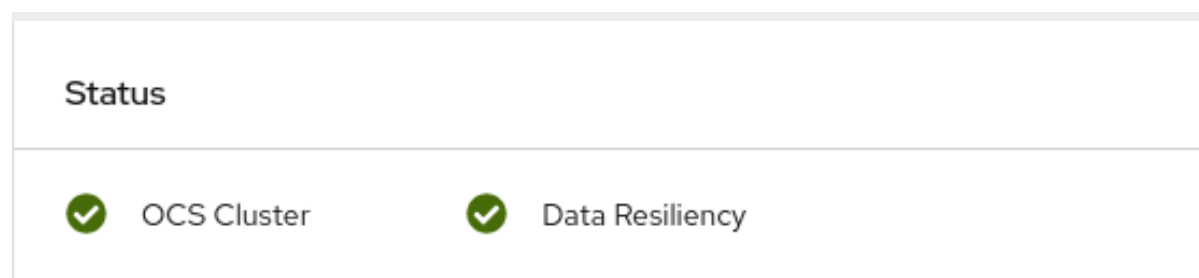
コンポーネント	対応する Pod
OpenShift Container Storage Operator	<ul style="list-style-type: none"> ● ocs-operator-* (任意のワーカーノードに 1 Pod) ● ocs-metrics-exporter-*
Rook-ceph Operator	rook-ceph-operator-* (任意のワーカーノードに 1 Pod)
Multicloud Object Gateway	<ul style="list-style-type: none"> ● noobaa-operator-* (任意のワーカーノードに 1 Pod) ● noobaa-core-* (任意のストレージノードに 1 Pod) ● noobaa-db-pg-* (任意のストレージノードに 1 Pod) ● noobaa-endpoint-* (任意のストレージノードに 1 Pod)
MON	rook-ceph-mon-* (ストレージノードに分散する 3 Pod)

コンポーネント	対応する Pod
MGR	rook-ceph-mgr-* (任意のストレージノードに 1 Pod)
MDS	rook-ceph-mds-ocs-storagecluster-cephfilesystem-* (ストレージノードに分散する 2 Pod)
CSI	<ul style="list-style-type: none"> ● cephfs <ul style="list-style-type: none"> ○ csi-cephfsplugin-* (各ワーカーノードに 1 Pod) ○ csi-cephfsplugin-provisioner-* (ワーカーノードに分散する 2 Pod) ● rbd <ul style="list-style-type: none"> ○ csi-rbdplugin-* (各ワーカーノードに 1 Pod) ○ csi-rbdplugin-provisioner-* (ストレージノードに分散する 2 Pod)
rook-ceph-crashcollector	rook-ceph-crashcollector-* (各ストレージノードに 1 Pod)
OSD	<ul style="list-style-type: none"> ● rook-ceph-osd-* (各デバイス用に 1 Pod) ● rook-ceph-osd-prepare-ocs-device-* (各デバイス用に 1 Pod)

4.2. OPENSIFT CONTAINER STORAGE クラスターが正常であることの確認

- OpenShift Web コンソールの左側のペインから **Home → Overview** をクリックし、**Persistent Storage** タブをクリックします。
- **Status** カードで、以下のイメージのように **OCS Cluster** および **Data Resiliency** に緑色のチェックマークが表示されていることを確認します。

図4.1 Persistent Storage Overview ダッシュボードの Health status カード



- **Details カード** で、以下のようにクラスター情報が表示されていることを確認します。

サービス名

OpenShift Container Storage

クラスター名

ocs-storagecluster

プロバイダー

AWS

モード

内部

バージョン

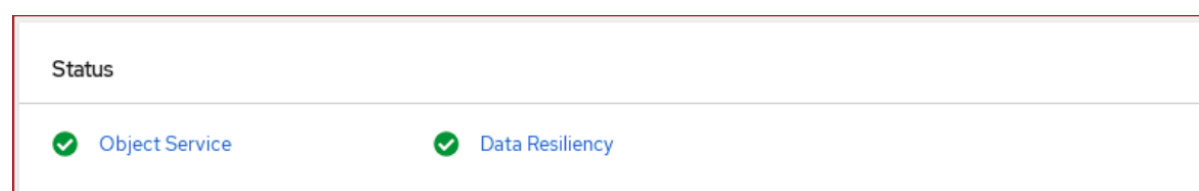
ocs-operator-4.7.0

永続ストレージダッシュボードを使用して OpenShift Container Storage クラスターの正常性に関する詳細は、[OpenShift Container Storage のモニターリング](#) を参照してください。

4.3. MULTICLOUD OBJECT GATEWAY が正常であることの確認

- OpenShift Web コンソールの左側のペインから **Home → Overview** をクリックし、**Object Service** タブをクリックします。
- **Status card** で、**Object Service** と **Data Resiliency** の両方が **Ready** 状態 (緑のチェックマーク) にあることを確認します。

図4.2 Object Service Overview ダッシュボードの Health status カード



- **Details カード** で、MCG 情報が以下のように表示されることを確認します。

サービス名

OpenShift Container Storage

システム名

Multicloud Object Gateway

プロバイダー

AWS

バージョン

ocs-operator-4.7.0

オブジェクトサービスダッシュボードを使用した OpenShift Container Storage クラスターの正常性については、[OpenShift Container Storage のモニターリング](#) を参照してください。

4.4. OPENSHIFT CONTAINER STORAGE 固有のストレージクラスが存在することの確認

ストレージクラスがクラスターに存在することを確認するには、以下を実行します。

- OpenShift Web コンソールの左側のペインから **Storage → Storage Classes** をクリックします。
- 以下のストレージクラスが OpenShift Container Storage クラスターの作成時に作成されることを確認します。
 - **ocs-storagecluster-ceph-rbd**
 - **ocs-storagecluster-cephfs**
 - **openshift-storage.noobaa.io**

第5章 OPENSIFT CONTAINER STORAGE のアンインストール

5.1. 内部モードでの OPENSIFT CONTAINER STORAGE のアンインストール

このセクションの手順に従って OpenShift Container Storage をアンインストールします。

アノテーションのアンインストール

Storage Cluster のアノテーションは、アンインストールプロセスの動作を変更するために使用されます。アンインストールの動作を定義するために、ストレージクラスターに以下の2つのアノテーションが導入されました。

- **uninstall.ocs.openshift.io/cleanup-policy: delete**
- **uninstall.ocs.openshift.io/mode: graceful**

以下の表は、これらのアノテーションで使用できる各種値に関する情報を示しています。

表5.1 **uninstall.ocs.openshift.io** でアノテーションの説明をアンインストールする

Annotation	値	デフォルト	動作
cleanup-policy	delete	はい	Rook は物理ドライブおよび DataDirHostPath をクリーンアップします。
cleanup-policy	Retain	いいえ	Rook は物理ドライブおよび DataDirHostPath をクリーンアップ しません 。
mode	graceful	はい	Rook および NooBaa は PVC および OBC が管理者/ユーザーによって削除されるまでアンインストールプロセスを一時停止します。
mode	forced	いいえ	Rook および NooBaa は、Rook および NooBaa を使用してプロビジョニングされた PVC/OBC がそれぞれ存在している場合でもアンインストールを続行します。

以下のコマンドを使用してアノテーションの値を編集し、クリーンアップポリシーまたはアンインストールモードを変更できます。

```
$ oc annotate storagecluster -n openshift-storage ocs-storagecluster
uninstall.ocs.openshift.io/cleanup-policy="retain" --overwrite
storagecluster.ocs.openshift.io/ocs-storagecluster annotated
```

```
$ oc annotate storagecluster -n openshift-storage ocs-storagecluster
uninstall.ocs.openshift.io/mode="forced" --overwrite
storagecluster.ocs.openshift.io/ocs-storagecluster annotated
```

前提条件

- OpenShift Container Storage クラスターの状態が正常であることを確認します。リソースまたはノードの不足により一部の Pod が正常に終了されないと、アンインストールプロセスに失敗する可能性があります。クラスターが状態が正常でない場合は、OpenShift Container Storage をアンインストールする前に Red Hat カスタマーサポートにお問い合わせください。
- アプリケーションが OpenShift Container Storage によって提供されるストレージクラスを使用して永続ボリューム要求 (PVC) またはオブジェクトバケット要求 (OBC) を使用していないことを確認します。
- カスタムリソース (カスタムストレージクラス、cephblockpools など) が管理者によって作成された場合、それらを消費したリソースを削除した後に管理者によって削除される必要があります。

手順

1. OpenShift Container Storage を使用しているボリュームスナップショットを削除します。

- a. すべての namespace からボリュームスナップショットを一覧表示します。

```
$ oc get volumesnapshot --all-namespaces
```

- b. 直前のコマンドの出力から、OpenShift Container Storage を使用しているボリュームスナップショットを特定し、削除します。

```
$ oc delete volumesnapshot <VOLUME-SNAPSHOT-NAME> -n <NAMESPACE>
```

2. OpenShift Container Storage を使用している PVC および OBC を削除します。
デフォルトのアンインストールモード (graceful) では、アンインストーラーは OpenShift Container Storage を使用するすべての PVC および OBC が削除されるまで待機します。

PVC を事前に削除せずに Storage Cluster を削除する場合は、アンインストールモードのアノテーションを forced に設定し、この手順を省略できます。これを実行すると、孤立した PVC および OBC がシステムに作成されます。

- a. OpenShift Container Storage を使用して、OpenShift Container Platform モニターリングスタック PVC を削除します。
[「OpenShift Container Storage からのモニターリングスタックの削除」](#) を参照してください。
- b. OpenShift Container Storage を使用して、OpenShift Container Platform レジストリー PVC を削除します。
[「OpenShift Container Storage からの OpenShift Container Platform レジストリーの削除」](#) を参照してください。

- c. OpenShift Container Storage を使用して、OpenShift Container Platform ロギング PVC を削除します。
「[OpenShift Container Storage からのクラスターロギング Operator の削除](#)」を参照してください。
- d. OpenShift Container Storage を使用してプロビジョニングした PVC および OBC を削除します。
 - 以下に、OpenShift Container Storage を使用してプロビジョニングされる PVC および OBC を特定するサンプルスクリプトを示します。このスクリプトは、OpenShift Container Storage によって内部で使用される PVC を無視します。

```
#!/bin/bash

RBD_PROVISIONER="openshift-storage.rbd.csi.ceph.com"
CEPHFS_PROVISIONER="openshift-storage.cephfs.csi.ceph.com"
NOOBAA_PROVISIONER="openshift-storage.noobaa.io/obc"
RGW_PROVISIONER="openshift-storage.ceph.rook.io/bucket"

NOOBAA_DB_PVC="noobaa-db"
NOOBAA_BACKINGSTORE_PVC="noobaa-default-backing-store-noobaa-pvc"

# Find all the OCS StorageClasses
OCS_STORAGECLASSES=$(oc get storageclasses | grep -e
"$RBD_PROVISIONER" -e "$CEPHFS_PROVISIONER" -e
"$NOOBAA_PROVISIONER" -e "$RGW_PROVISIONER" | awk '{print $1}')

# List PVCs in each of the StorageClasses
for SC in $OCS_STORAGECLASSES
do
    echo
    "=====
=="
    echo "$SC StorageClass PVCs and OBCs"
    echo
    "=====
=="
    oc get pvc --all-namespaces --no-headers 2>/dev/null | grep $SC | grep -v -e
"$NOOBAA_DB_PVC" -e "$NOOBAA_BACKINGSTORE_PVC"
    oc get obc --all-namespaces --no-headers 2>/dev/null | grep $SC
    echo
done
```



注記

クラウドプラットフォームの **RGW_PROVISIONER** を省略します。

- OBC を削除します。

```
$ oc delete obc <obc name> -n <project name>
```

- PVC を削除します。

```
$ oc delete pvc <pvc name> -n <project-name>
```




注記

クラスターに作成されているカスタムバックリングストア、バケットクラスなどを削除していることを確認します。

- Storage Cluster オブジェクトを削除し、関連付けられたリソースが削除されるのを待機します。

```
$ oc delete -n openshift-storage storagecluster --all --wait=true
```

- uninstall.ocs.openshift.io/cleanup-policy** が **delete** (default) に設定されている場合にクリーンアップ Pod の有無を確認し、それらのステータスが **Completed** していることを確認します。

```
$ oc get pods -n openshift-storage | grep -i cleanup
NAME                                READY STATUS RESTARTS AGE
cluster-cleanup-job-<xx>            0/1   Completed 0      8m35s
cluster-cleanup-job-<yy>            0/1   Completed 0      8m35s
cluster-cleanup-job-<zz>            0/1   Completed 0      8m35s
```

- /var/lib/rook** ディレクトリーが空であることを確認します。このディレクトリーは空になるのは、**uninstall.ocs.openshift.io/cleanup-policy** アノテーションが **delete** (デフォルト) に設定されている場合に限られます。

```
$ for i in $(oc get node -l cluster.ocs.openshift.io/openshift-storage= -o jsonpath='{.items[*].metadata.name}'); do oc debug node/${i} -- chroot /host ls -l /var/lib/rook; done
```

- 暗号化がインストール時に有効にされている場合は、すべての OpenShift Container Storage ノードの OSD デバイスから **dm-crypt** で管理される **device-mapper** マッピングを削除します。

- デバッグ** Pod を作成し、ストレージノードのホストに対して **chroot** を作成します。

```
$ oc debug node/<node name>
$ chroot /host
```

- デバイス名を取得し、OpenShift Container Storage デバイスについてメモします。

```
$ dmsetup ls
ocs-deviceset-0-data-0-57snx-block-dmccrypt (253:1)
```

- マップ済みデバイスを削除します。

```
$ cryptsetup luksClose --debug --verbose ocs-deviceset-0-data-0-57snx-block-dmccrypt
```



注記

権限が十分でないため、コマンドがスタックした場合には、以下のコマンドを実行します。

- **CTRL+Z** を押して上記のコマンドを終了します。
- スタックしたプロセスの PID を検索します。

```
$ ps -ef | grep crypt
```

- **kill** コマンドを使用してプロセスを終了します。

```
$ kill -9 <PID>
```

- デバイス名が削除されていることを確認します。

```
$ dmsetup ls
```

7. namespace を削除し、削除が完了するまで待機します。**openshift-storage** がアクティブなプロジェクトである場合は、別のプロジェクトに切り替える必要があります。以下に例を示します。

```
$ oc project default
$ oc delete project openshift-storage --wait=true --timeout=5m
```

以下のコマンドが NotFound エラーを返すと、プロジェクトが削除されます。

```
$ oc get project openshift-storage
```



注記

OpenShift Container Storage のアンインストール時に、**namespace** が完全に削除されず、**Terminating** 状態のままである場合は、[Troubleshooting and deleting remaining resources during Uninstall](#) の記事に記載の手順を実行して namespace の終了をブロックしているオブジェクトを特定します。

8. ローカルストレージデバイスを使用して OpenShift Container Storage をデプロイした場合には、ローカルのストレージ Operator 設定を削除します。[ローカルストレージ Operator の設定の削除](#) を参照してください。
9. ストレージノードのラベルを解除します。

```
$ oc label nodes --all cluster.ocs.openshift.io/openshift-storage-
$ oc label nodes --all topology.rook.io/rack-
```

10. ノードにテイントのマークが付けられている場合に OpenShift Container Storage テイントを削除します。

```
$ oc adm taint nodes --all node.ocs.openshift.io/storage-
```

11. OpenShift Container Storage を使用してプロビジョニングした PV がすべて削除されていることを確認します。**Released** 状態のままの PV がある場合は、これを削除します。

```
$ oc get pv
$ oc delete pv <pv name>
```

12. Multicloud Object Gateway storageclass を削除します。

```
$ oc delete storageclass openshift-storage.noobaa.io --wait=true --timeout=5m
```

13. **CustomResourceDefinitions** を削除します。

```
$ oc delete crd backingstores.noobaa.io bucketclasses.noobaa.io
cephblockpools.ceph.rook.io cephclusters.ceph.rook.io cephfilesystems.ceph.rook.io
cephnfses.ceph.rook.io cephobjectstores.ceph.rook.io cephobjectstoreusers.ceph.rook.io
noobaas.noobaa.io ocsinitializations.ocs.openshift.io storageclusters.ocs.openshift.io
cephclients.ceph.rook.io cephobjectrealms.ceph.rook.io cephobjectzonegroups.ceph.rook.io
cephobjectzones.ceph.rook.io cephrbdmirrors.ceph.rook.io --wait=true --timeout=5m
```

14. オプション: vault キーが完全に削除されるようにするには、vault キーに関連付けられているメタデータを手動で削除する必要があります。



注記

このステップは、Vault Key/Value (KV) シークレットエンジン API バージョン 2 が、OpenShift Container Storage のアンインストール中に Vault キーが削除済みとしてマークされ、完全に削除されないため、Key Management System (KMS) によるクラスター全体の暗号化に使用される場合にのみ実行してください。必要に応じて、後でいつでも復元できます。

- a. Vault 内のキーを一覧表示します。

```
$ vault kv list <backend_path>
```

<backend_path>

暗号化キーが保存されている Vault 内のパスです。
以下に例を示します。

```
$ vault kv list kv-v2
```

出力例:

```
Keys
-----
NOOBAA_ROOT_SECRET_PATH/
rook-ceph-osd-encryption-key-ocs-deviceset-thin-0-data-0m27q8
rook-ceph-osd-encryption-key-ocs-deviceset-thin-1-data-0sq227
rook-ceph-osd-encryption-key-ocs-deviceset-thin-2-data-0xzszb
```

- b. Vault キーに関連付けられているメタデータを一覧表示します。

```
$ vault kv get kv-v2/<key>
```

■

Multicloud Object Gateway (MCG) キーの場合:

```
$ vault kv get kv-v2/NOOBAA_ROOT_SECRET_PATH/<key>
```

<key>

暗号化キーです。
以下に例を示します。

```
$ vault kv get kv-v2/rook-ceph-osd-encryption-key-ocs-deviceset-thin-0-data-0m27q8
```

出力例:

```
===== Metadata =====
Key          Value
---          -
created_time  2021-06-23T10:06:30.650103555Z
deletion_time 2021-06-23T11:46:35.045328495Z
destroyed     false
version       1
```

c. メタデータを削除します。

```
$ vault kv metadata delete kv-v2/<key>
```

MCG キーの場合:

```
$ vault kv metadata delete kv-v2/NOOBAA_ROOT_SECRET_PATH/<key>
```

<key>

暗号化キーです。
以下に例を示します。

```
$ vault kv metadata delete kv-v2/rook-ceph-osd-encryption-key-ocs-deviceset-thin-0-
data-0m27q8
```

出力例:

```
Success! Data deleted (if it existed) at: kv-v2/metadata/rook-ceph-osd-encryption-key-
ocs-deviceset-thin-0-data-0m27q8
```

d. これらの手順を繰り返して、すべての Vault キーに関連付けられているメタデータを削除します。

15. OpenShift Container Platform Web コンソールで、OpenShift Container Storage が完全にアンインストールされていることを確認するには、以下を実行します。

a. **Home → Overview** をクリックし、ダッシュボードにアクセスします。

b. Persistent Storage および Object Service タブが **Cluster** タブの横に表示されなくなることを確認します。

5.1.1. ローカルストレージ Operator の設定の削除

ローカルストレージデバイスを使用して OpenShift Container Storage をデプロイした場合にのみ、本セクションの手順を使用します。



注記

OpenShift Container Storage デプロイメントで **localvolume** リソースのみを使用する場合は、直接、手順 8 に移動します。

手順

1. **LocalVolumeSet** および OpenShift Container Storage で使用される対応する **StorageClassName** を特定します。

```
$ export SC="<StorageClassName>"
```

3. **LocalVolumeSet** を削除します。

```
$ oc delete localvolumesets.local.storage.openshift.io <name-of-volumeset> -n openshift-local-storage
```

4. 指定された **StorageClassName** のローカルストレージ PV を削除します。

```
$ oc get pv | grep $SC | awk '{print $1}' | xargs oc delete pv
```

5. **StorageClassName** を削除します。

```
$ oc delete sc $SC
```

6. **LocalVolumeSet** によって作成されるシンボリックリンクを削除します。

```
[[ ! -z $SC ]] && for i in $(oc get node -l cluster.ocs.openshift.io/openshift-storage= -o jsonpath='{.items[*].metadata.name}'); do oc debug node/${i} -- chroot /host rm -rfv /mnt/local-storage/${SC}; done
```

7. **LocalVolumeDiscovery** を削除します。

```
$ oc delete localvolumediscovery.local.storage.openshift.io/auto-discover-devices -n openshift-local-storage
```

8. **LocalVolume** リソースを削除します (ある場合)。

以下の手順を使用して、現行または直前の OpenShift Container Storage バージョンで PV のプロビジョニングに使用した **LocalVolume** リソースを削除します。また、これらのリソースがクラスターの他のテナントで使用されていないことを確認します。

ローカルボリュームごとに、以下を実行します。

- a. **LocalVolume** および OpenShift Container Storage で使用される対応する **StorageClassName** を特定します。

- b. 変数 LV を LocalVolume の名前に設定し、変数 SC を StorageClass の名前に設定します。以下に例を示します。

```
$ LV=local-block
$ SC=localblock
```

- c. ローカルボリュームリソースを削除します。

```
$ oc delete localvolume -n openshift-local-storage --wait=true $LV
```

- d. 残りの PV および StorageClass が存在する場合はこれらを削除します。

```
$ oc delete pv -l storage.openshift.com/local-volume-owner-name=${LV} --wait --
timeout=5m
$ oc delete storageclass $SC --wait --timeout=5m
```

- e. そのリソースのストレージノードからアーティファクトをクリーンアップします。

```
$ [[ ! -z $SC ]] && for i in $(oc get node -l cluster.ocs.openshift.io/openshift-storage= -o
jsonpath='{.items[*].metadata.name}'); do oc debug node/${i} -- chroot /host rm -rfv
/mnt/local-storage/${SC}/; done
```

出力例:

```
Starting pod/node-xxx-debug ...
To use host binaries, run `chroot /host`
removed '/mnt/local-storage/localblock/nvme2n1'
removed directory '/mnt/local-storage/localblock'
```

```
Removing debug pod ...
Starting pod/node-yyy-debug ...
To use host binaries, run `chroot /host`
removed '/mnt/local-storage/localblock/nvme2n1'
removed directory '/mnt/local-storage/localblock'
```

```
Removing debug pod ...
Starting pod/node-zzz-debug ...
To use host binaries, run `chroot /host`
removed '/mnt/local-storage/localblock/nvme2n1'
removed directory '/mnt/local-storage/localblock'
```

```
Removing debug pod ...
```

5.2. OPENSIFT CONTAINER STORAGE からのモニターリングスタックの削除

このセクションでは、モニターリングスタックを OpenShift Container Storage からクリーンアップします。

モニターリングスタックの設定の一部として作成される PVC は **openshift-monitoring** namespace に置かれます。

前提条件

- PVC は OpenShift Container Platform モニタリングスタックを使用できるように設定されます。
詳細は、[モニタリングスタックの設定](#) を参照してください。

手順

1. **openshift-monitoring** namespace で現在実行されている Pod および PVC を一覧表示します。

```
$ oc get pod,pvc -n openshift-monitoring
```

NAME	READY	STATUS	RESTARTS	AGE
pod/alertmanager-main-0	3/3	Running	0	8d
pod/alertmanager-main-1	3/3	Running	0	8d
pod/alertmanager-main-2	3/3	Running	0	8d
pod/cluster-monitoring-operator-84457656d-pkrxm	1/1	Running	0	8d
pod/grafana-79ccf6689f-2ll28	2/2	Running	0	8d
pod/kube-state-metrics-7d86fb966-rvd9w	3/3	Running	0	8d
pod/node-exporter-25894	2/2	Running	0	8d
pod/node-exporter-4dsd7	2/2	Running	0	8d
pod/node-exporter-6p4zc	2/2	Running	0	8d
pod/node-exporter-jbjvg	2/2	Running	0	8d
pod/node-exporter-jj4t5	2/2	Running	0	6d18h
pod/node-exporter-k856s	2/2	Running	0	6d18h
pod/node-exporter-rf8gn	2/2	Running	0	8d
pod/node-exporter-rmb5m	2/2	Running	0	6d18h
pod/node-exporter-zj7kx	2/2	Running	0	8d
pod/openshift-state-metrics-59dbd4f654-4clng	3/3	Running	0	8d
pod/prometheus-adapter-5df5865596-k8dzn	1/1	Running	0	7d23h
pod/prometheus-adapter-5df5865596-n2gj9	1/1	Running	0	7d23h
pod/prometheus-k8s-0	6/6	Running	1	8d
pod/prometheus-k8s-1	6/6	Running	1	8d
pod/prometheus-operator-55cfb858c9-c4zd9	1/1	Running	0	6d21h
pod/telemeter-client-78fc8fc97d-2rgfp	3/3	Running	0	8d

NAME	CAPACITY	ACCESS MODES	STATUS	VOLUME STORAGECLASS	AGE
persistentvolumeclaim/my-alertmanager-claim-alertmanager-main-0	40Gi	Bound	RWO	pvc-0d519c4f-15a5-11ea-baa0-026d231574aa	8d
persistentvolumeclaim/my-alertmanager-claim-alertmanager-main-1	40Gi	Bound	RWO	pvc-0d5a9825-15a5-11ea-baa0-026d231574aa	8d
persistentvolumeclaim/my-alertmanager-claim-alertmanager-main-2	40Gi	Bound	RWO	pvc-0d6413dc-15a5-11ea-baa0-026d231574aa	8d
persistentvolumeclaim/my-prometheus-claim-prometheus-k8s-0	40Gi	Bound	RWO	pvc-0b7c19b0-15a5-11ea-baa0-026d231574aa	8d
persistentvolumeclaim/my-prometheus-claim-prometheus-k8s-1	40Gi	Bound	RWO	pvc-0b8aed3f-15a5-11ea-baa0-026d231574aa	8d

2. モニタリング **configmap** を編集します。

```
$ oc -n openshift-monitoring edit configmap cluster-monitoring-config
```

3. 以下の例が示すように、OpenShift Container Storage ストレージクラスを参照する **config** セクションを削除し、これを保存します。

編集前

```
.
.
.
apiVersion: v1
data:
  config.yaml: |
    alertmanagerMain:
      volumeClaimTemplate:
        metadata:
          name: my-alertmanager-claim
        spec:
          resources:
            requests:
              storage: 40Gi
          storageClassName: ocs-storagecluster-ceph-rbd
    prometheusK8s:
      volumeClaimTemplate:
        metadata:
          name: my-prometheus-claim
        spec:
          resources:
            requests:
              storage: 40Gi
          storageClassName: ocs-storagecluster-ceph-rbd
kind: ConfigMap
metadata:
  creationTimestamp: "2019-12-02T07:47:29Z"
  name: cluster-monitoring-config
  namespace: openshift-monitoring
  resourceVersion: "22110"
  selfLink: /api/v1/namespaces/openshift-monitoring/configmaps/cluster-monitoring-config
  uid: fd6d988b-14d7-11ea-84ff-066035b9efa8
.
```

編集後


```

.
.
.
apiVersion: v1
data:
  config.yaml: |
kind: ConfigMap
metadata:
  creationTimestamp: "2019-11-21T13:07:05Z"
  name: cluster-monitoring-config
  namespace: openshift-monitoring
  resourceVersion: "404352"
  selfLink: /api/v1/namespaces/openshift-monitoring/configmaps/cluster-monitoring-config
  uid: d12c796a-0c5f-11ea-9832-063cd735b81c
.
.
.

```

この例では、**alertmanagerMain** および **prometheusK8s** モニターリングコンポーネントは OpenShift Container Storage PVC を使用しています。

4. 関連する PVC を削除します。ストレージクラスを使用するすべての PVC を削除してください。

```
$ oc delete -n openshift-monitoring pvc <pvc-name> --wait=true --timeout=5m
```

5.3. OPENSIFT CONTAINER STORAGE からの OPENSIFT CONTAINER PLATFORM レジストリーの削除

このセクションを使用して、OpenShift Container Storage から OpenShift Container Platform レジストリーをクリーンアップします。代替ストレージを設定する必要がある場合、[イメージレジストリー](#) を参照してください。

OpenShift Container Platform レジストリーの設定の一部として作成される PVC は **openshift-image-registry** namespace に置かれます。

前提条件

- イメージレジストリーは OpenShift Container Storage PVC を使用するように設定されている必要があります。

手順

1. **configs.imageregistry.operator.openshift.io** オブジェクトを編集し、**storage** セクションのコンテンツを削除します。

```
$ oc edit configs.imageregistry.operator.openshift.io
```

編集前

```

.
.
.
storage:
  pvc:
    claim: registry-cephfs-rwx-pvc
.
.
.

```

編集後

```

.
.
.
storage:
.
.
.

```

この例では、PVC は **registry-cephfs-rwx-pvc** と呼ばれ、これは安全に削除できます。

2. PVC を削除します。

```
$ oc delete pvc <pvc-name> -n openshift-image-registry --wait=true --timeout=5m
```

5.4. OPENSIFT CONTAINER STORAGE からのクラスターロギング OPERATOR の削除

このセクションでは、クラスターロギング Operator を OpenShift Container Storage からクリーンアップします。

クラスターロギング Operator の設定の一部として作成される PVC は **openshift-logging** namespace にあります。

前提条件

- クラスターロギングインスタンスは、OpenShift Container Storage PVC を使用するよう設定されている必要があります。

手順

1. namespace の **ClusterLogging** インスタンスを削除します。

```
$ oc delete clusterlogging instance -n openshift-logging --wait=true --timeout=5m
```

openshift-logging namespace の PVC は安全に削除できます。

2. PVC を削除します。

```
$ oc delete pvc <pvc-name> -n openshift-logging --wait=true --timeout=5m
```