Red Hat OpenShift Container Storage 4.5

ベアメタルインフラストラクチャーを使用した OpenShift Container Storage のデプロイ

ベアメタル環境のインストールおよび設定方法

Last Updated: 2023-03-28

Red Hat OpenShift Container Storage 4.5 ベアメタルインフラストラクチャーを使用した OpenShift Container Storage のデプロイ

ベアメタル環境のインストールおよび設定方法

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

概要

Red Hat OpenShift Container Storage 4.5 をインストールし、ベアメタルインフラストラクチャーでローカルストレージを使用する方法については、本書をお読みください。

目次

はじめに

Red Hat OpenShift Container Storage 4.5 は、接続環境または非接続環境での既存の Red Hat OpenShift Container Platform (OCP) ベアメタルクラスターへのデプロイメントをサポートし、プロキシー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

ベアメタルでは、内部と外部の両方の Openshift Container Storage クラスター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、『デプロイメントのプランニング』を参照してください。

OpenShift Container Storage をデプロイするには、お使いの環境に適切なデプロイメントプロセスを実行します。

- 内部モード
  - ローカルストレージデバイスを使用したデプロイ
- 外部モード

第1章 ローカルストレージデバイスを使用したデプロイメント

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ベアメタルインフラストラクチャーに OpenShift Container Storage をインストールします。

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. 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。
2. 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ー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。

3. Red Hat OpenShift Container Storage Operator をインストールします 。
4. ローカルストレージ Operator をインストールします。
5. 利用可能なストレージデバイスを見つけます。
6. ベアメタルで OpenShift Container Storage クラスターサービスを作成します 。

1.1. ローカルストレージデバイスを使用した OPENSHIFT CONTAINER STORAGE のインストール要件

- クラスターに、それぞれローカルで割り当てられたストレージデバイスを持つ OpenShift Container Platform ワーカーノードを 3 つ以上設定する必要があります。
  - 3 つのノードのそれぞれには、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。
  - 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。
  - 使用するデバイスは空である必要があります。つまり、ディスクには PV、VG、または LV がない状態でなければなりません。
- 3 つ以上のラベルが付けられたノードが必要です。
  - 高可用性を確保するために、ワーカーノードは 3 つの異なる物理ノード、ラック、障害ドメインに分散することが推奨されます。
  - 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=''
```

- Red Hat OpenShift Container Storage のローカルストレージ Operator の使用と競合するストレージノードでローカルにマウントされたストレージを管理するストレージプロバイダーは使用しないでください。
- ローカルストレージ Operator が Red Hat OpenShift Container Storage で完全にサポートされるために、ローカルストレージ Operator のバージョンは Red Hat OpenShift Container Platform バージョンと一致する必要があります。ローカルストレージ Operator は、Red Hat OpenShift Container Platform のアップグレード時にアップグレードされません。

1.2. RED HAT ENTERPRISE LINUX ベースのノード上のコンテナー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) の Red Hat Enterprise Linux ベースに OpenShift Container Platform をデプロイしても、基礎となる Ceph ファイルシステムへのコンテナーアクセスは自動的に提供されません。

注記

このプロセスは、Red Hat Enterprise Linux CoreOS をベースとするホストには不要です。

手順

クラスター内の各ノードで以下の手順を実行します。

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. ノードが rhel-7-server-extras-rpms リポジトリー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ーを有効にします。

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

3. 必要なパッケージをインストールします。

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

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

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

1.3. RED HAT OPENSHIFT 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。ハードウェアおよびソフトウェアの要件に関する詳細は、『デプロイメントのプランニング』を参照してください。

前提条件

- OpenShift Container Platform クラスターにログインしている必要がある。
- OpenShift Container Platform クラスターにワーカーノードが少なくとも 3 つ必要です。

注記

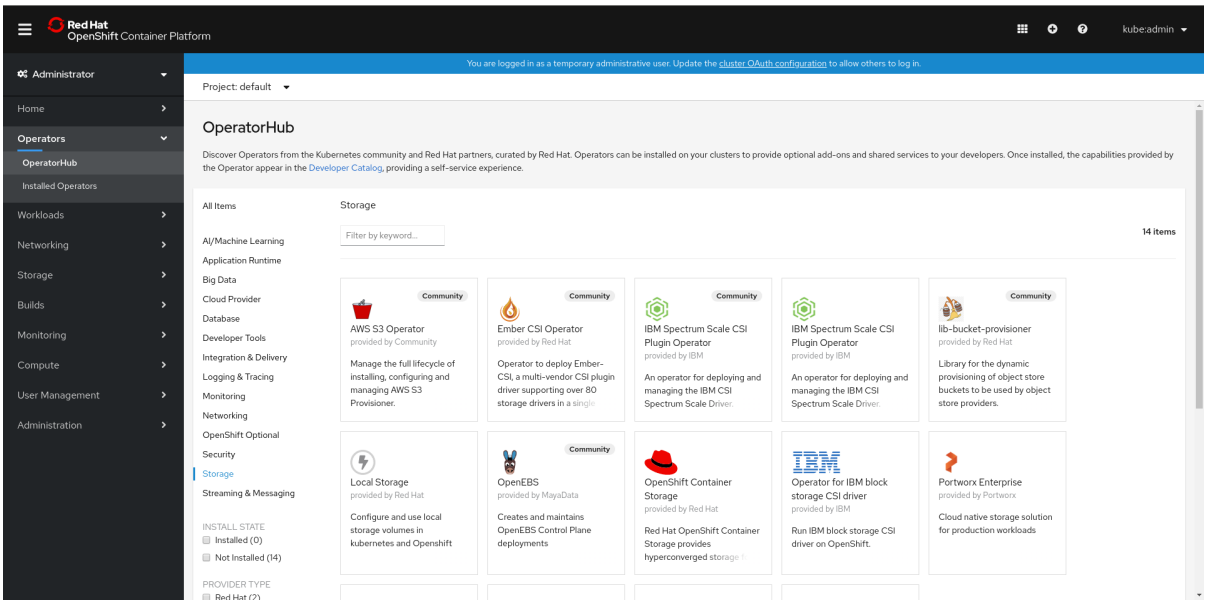
OpenShift Container Storage のクラスター全体でのデフォルトノードセレクターを上書きする必要がある場合は、コマンドラインインターフェースで以下のコマンドを使用し、**openshift-storage** namespace の空のノードセレクターを指定できます。

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

手順

1. OpenShift Web コンソールの左側のペインで、**Operators → OperatorHub** をクリックします。

図1.1 Operator Hub の Operator 一覧



2. **OpenShift Container Storage** をクリックします。
Filter by keyword テキストボックスまたはフィルター一覧を使用して、Operator の一覧から OpenShift Container Storage を検索できます。

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、以下のオプションが選択されていることを確認します。

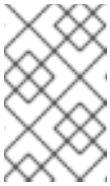
   a. Channel を **stable-4.5**として更新します。

   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 PR 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** として選択している。承認ストラテジーはデフォルトで **Automatic** に設定されます。

- **Approval Strategy** に **Automatic** を選択します。



注記

Approval Strategy を **Automatic** として選択すると、新規インストール時、または OpenShift Container Storage の最新バージョンへの更新時に承認は必要ありません。

i. **インストール** をクリックします。

ii. インストールが開始するまで待機します。これには、最長 20 分の時間がかかる可能性があります。

iii. **Operators → Installed Operators** をクリックします。

iv. **Project** が **openshift-storage** であることを確認します。デフォルトで、**プロジェクト** は **openshift-storage** です。

v. **OpenShift Container Storage** の **Status** が **Succeeded** に変更するまで待機します。

- **Approval Strategy** に **Manual** を選択します。

注記

Approval Strategy を **Manual** として選択すると、新規インストール時、または OpenShift Container Storage の最新バージョンへの更新時に承認が必要になります。

i. **Install** をクリックします。

ii. **Installed Operators** ページで、**ocs-operator** をクリックします。

iii. **Subscription Details** ページで、**Install Plan** リンクをクリックします。

iv. **InstallPlan Details** ページで、**Preview Install Plan** をクリックします。

v. インストール計画を確認し、**Approve** をクリックします。

vi. **Components** の **Status** が **Unknown** から **Created** または **Present** のいずれかに変更するまで待機します。

vii. **Operators → Installed Operators** をクリックします。

viii. **Project** が **openshift-storage** であることを確認します。デフォルトで、**プロジェクト** は **openshift-storage** です。

ix. **OpenShift Container Storage** の **Status** が **Succeeded** に変更するまで待機します。

検証手順

- OpenShift Container Storage Operator の Status が Installed Operators ダッシュボードで **Succeeded** と表示されることを確認します。

1.4. ローカルストレージ OPERATOR のインストール

以下の手順を使用して、OpenShift Container Storage クラスターをローカルストレージデバイスに作成する前に Operator Hub からローカルストレージ Operator をインストールします。

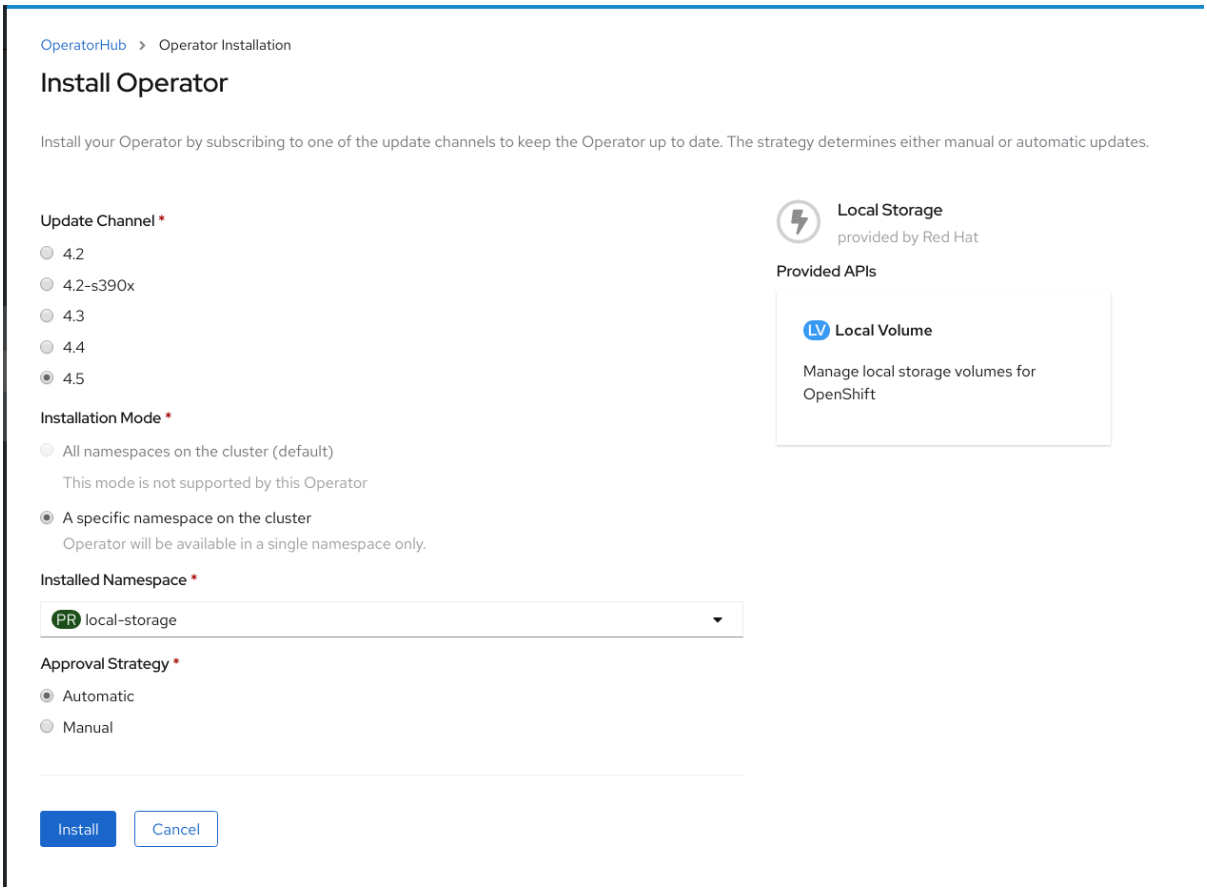
前提条件

- 以下のように、**openshift-storage** という namespace を作成します。
  a. OpenShift Web コンソールの左側のペインで、**Administration → Namespaces** をクリックします。
  b. **Create Namespace** をクリックします。
  c. Create Namespace ダイアログボックスで、Name に **local-storage** と入力します。
  d. **Default Network Policy** に **No restrictions** オプションを選択します。
  e. **Create** をクリックします。

手順

1. OpenShift Web コンソールの左側のペインで、**Operators → OperatorHub** をクリックします。
2. Operator の一覧から **Local Storage Operator** を検索し、これをクリックします。
3. **Install** をクリックします。

図1.2 Install Operator ページ



4. **Install Operator** ページで、以下のオプションが選択されていることを確認します。
   a. Channel を **stable-4.5**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 を **local-storage** に選択します。
   d. Approval Strategy に **Automatic** を選択します。
5. **Install** をクリックします。
6. Local Storage Operator のステータスが **Succeeded** と表示されていることを確認します。

1.5. 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、ベアメタルの 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 つ以上のワーカー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 ラベルの付いたワーカー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
bmworker01   Ready    worker   6h45m  v1.16.2
bmworker02   Ready    worker   6h45m  v1.16.2
bmworker03   Ready    worker   6h45m  v1.16.2
```

2. OpenShift Container Storage リソースに使用される各ワーカー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/<Nodename>
```

出力例:

```
$ oc debug node/bmworker01
Starting pod/bmworker01-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   931G  0 disk
```

この例では、**bmworker01** の場合、利用可能なローカルデバイスは **nvme0n1** です。

3. 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 nvme0n1
lrwxrwxrwx. 1 root root 13 Mar 17 16:24 nvme-
INTEL_SSDPE2KX010T7_PHLF733402LM1P0GGN -> ../../nvme0n1
```

上記の例では、ローカルデバイス **nvme0n1**の ID

```
nvme-INTEL_SSDPE2KX010T7_PHLF733402LM1P0GGN
```

4. 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ティクルを参照してください。

1.6. ベアメタルでの OPENSHIFT CONTAINER STORAGE クラスターの作成

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 についてのセクションにあるすべての要件を満たしていることを確認します。

- ベアメタルでローカルストレージデバイスを使用するために、同じストレージタイプおよびサイズが各ノードに割り当てられた 3 つのワーカーノードが必要です (例: 2TB NVMe ハードドライブ)。
- OpenShift Container Platform ワーカーノードに OpenShift Contaner Storage ラベルを付けられていることを確認します。

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

各ノードのストレージデバイスを特定するには、利用可能なストレージデバイスの検索について参照してください。

手順

1. ブロック PV の **LocalVolume** CR を作成します。
OCS ラベルをノードセレクターとして使用する **LocalVolume** CR **local-storage-block.yaml** の例。

```
apiVersion: local.storage.openshift.io/v1
kind: LocalVolume
metadata:
  name: local-block
  namespace: local-storage
  labels:
    app: ocs-storagecluster
spec:
  nodeSelector:
    nodeSelectorTerms:
    - matchExpressions:
        - key: cluster.ocs.openshift.io/openshift-storage
          operator: In
          values:
          - ""
  storageClassDevices:
    - storageClassName: localblock
      volumeMode: Block
      devicePaths:
        - /dev/disk/by-id/nvme-INTEL_SSDPEKKA128G7_BTPY81260978128A   # <-- modify
this line
        - /dev/disk/by-id/nvme-INTEL_SSDPEKKA128G7_BTPY80440W5U128A   # <-- modify
this line
        - /dev/disk/by-id/nvme-INTEL_SSDPEKKA128G7_BTPYB85AABDE128A   # <-- modify
this line
```

2. ブロック PV の **LocalVolume** CR を作成します。

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

3. Pod が作成されているかどうかを確認します。
出力例:

```
NAME                                READY
local-block-local-diskmaker-cmfql       1/1
```

```
local-block-local-diskmaker-g6fzr       1/1
local-block-local-diskmaker-jkqxt       1/1
local-block-local-provisioner-jgqcc     1/1
local-block-local-provisioner-mx49d     1/1
local-block-local-provisioner-qbcvp     1/1
local-storage-operator-54bc7566c6-ddbrt 1/1

STATUS  RESTARTS AGE
Running 0       31s
Running 0       31s
Running 0       31s
Running 0       31s
Running 0       31s
Running 0       31s
Running 0       12m
```

4. PV が作成されているかどうかを確認します。

```
$ oc get pv
```

出力例:

```
NAME                CAPACITY   ACCESS MODES   RECLAIM POLICY
local-pv-150fdc87   931Gi      RWO            Delete
local-pv-183bfc0a   931Gi      RWO            Delete
local-pv-b2f5cb25   931Gi      RWO            Delete

STATUS      CLAIM   STORAGECLASS   REASON   AGE
Available           localblock             2m11s
Available           localblock             2m15s
Available           localblock             2m21s
```

5. 新規 **localblock StorageClass** を確認します。

```
$ oc get sc|egrep -e "localblock|NAME"
```

出力例:

```
NAME       PROVISIONER                    RECLAIMPOLICY
localblock kubernetes.io/no-provisioner Delete

VOLUMEBINDINGMODE    ALLOWVOLUMEEXPANSION AGE
WaitForFirstConsumer false                4d23h
```

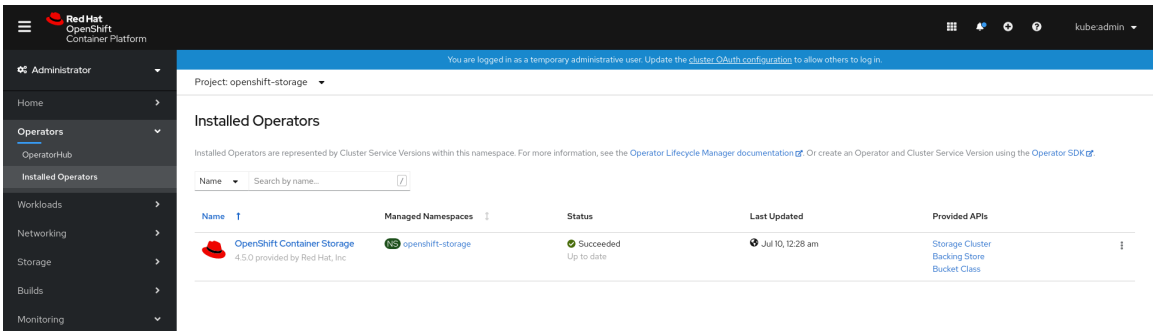
6. **localblock** Storage Class を使用する OpenShift Container Storage Cluster Service を作成します。

   a. OpenShift Web コンソールにログインします。

   b. OpenShift Web コンソールから **Operators → Installed Operators** をクリックし、インストールされた Operator を表示します。選択された **Project** が **openshift-storage** であることを確認します。

   c. **Installed Operators** ページで、**Openshift Container Storage** をクリックします。

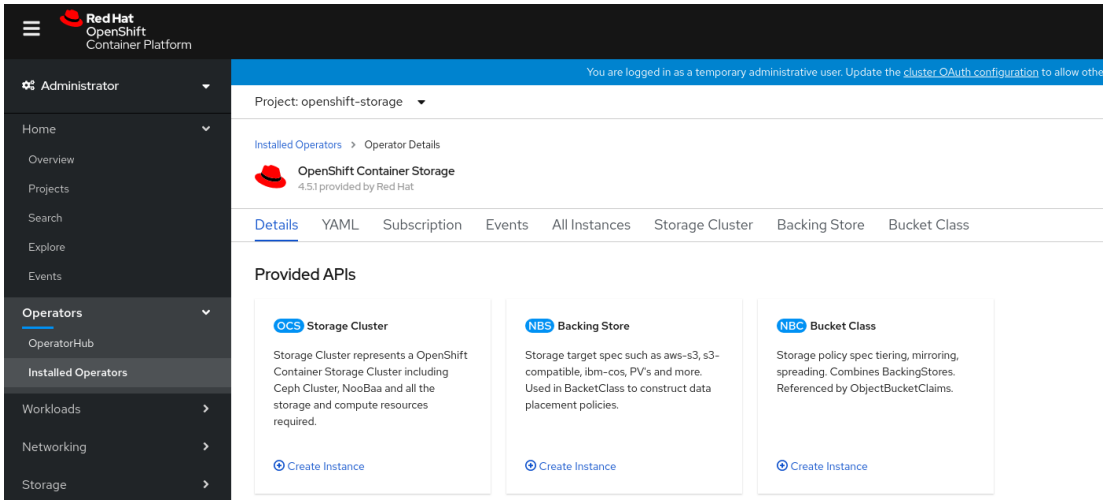
図1.3 OpenShift Container Storage Operator ページ



d. **Installed Operators → Operator Details**ページで、以下のいずれかを実行して Storage Cluster Service を作成します。

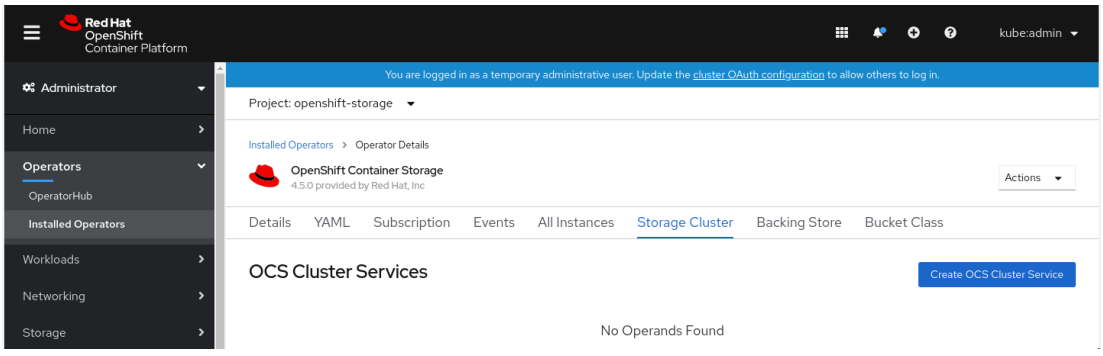
- **Details タブで Provided APIs → OCS Storage Cluster**で、**Create Instance** をクリックします。

図1.4 Operator Details ページ

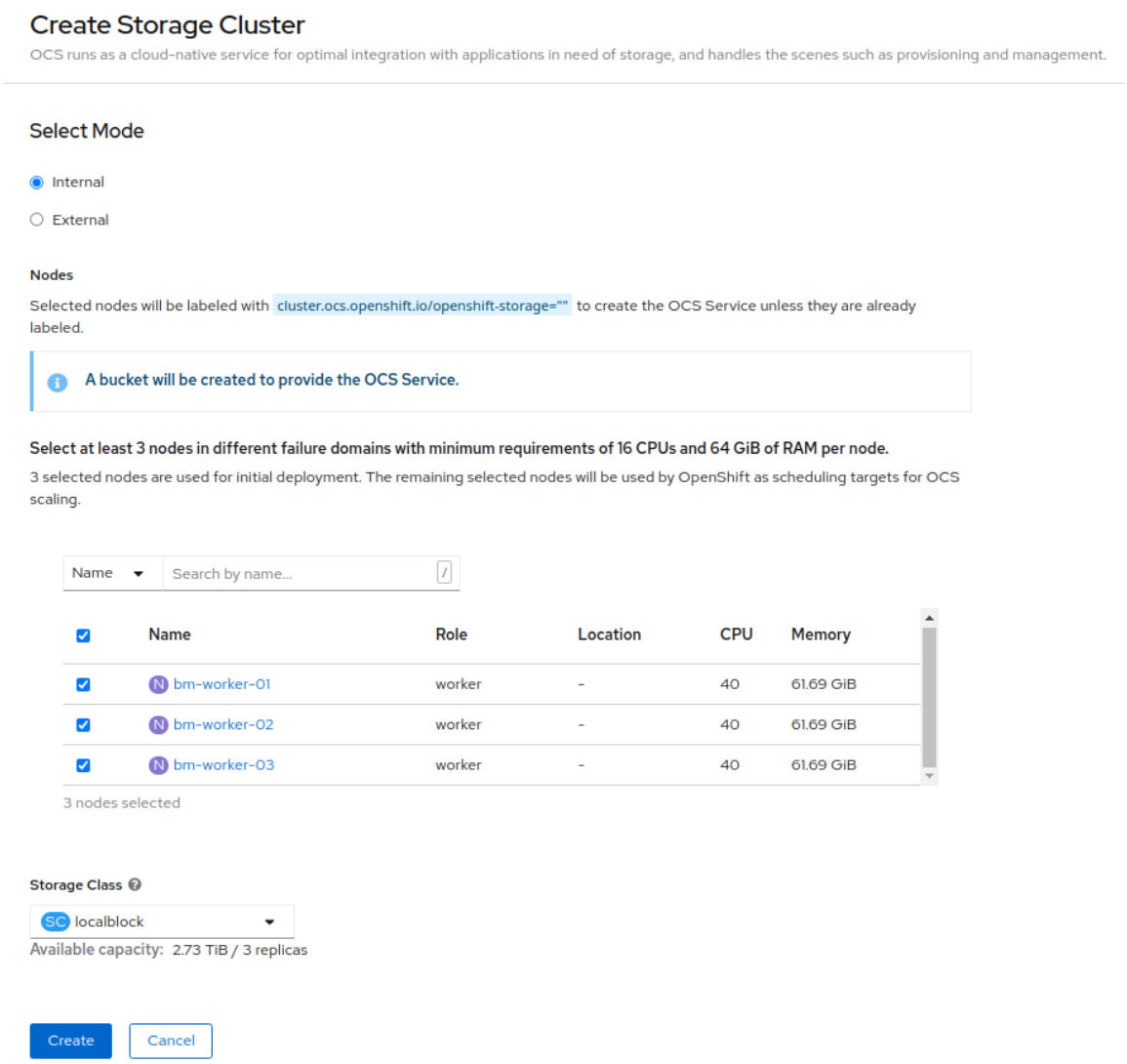


- または、**Storage cluster** タブを選択し、**Create OCS Cluster Service**をクリックします。

図1.5 Storage Cluster タブ



e. **Create Storage Cluster**ページで、以下のオプションが選択されていることを確認します。



- **Select Mode** を **Internal** のままにします。
- **Nodes** セクションでは、OpenShift Container Storage サービスを使用するには、利用可能な一覧から 3 つ以上のワーカーノードを選択します。
  高可用性を確保するために、ワーカーノードは 3 つの異なる物理ノード、ラック、障害ドメインに分散することが推奨されます。

注記

  - クラスターで特定のワーカーノードを見つけるには、Name または Label に基づいてノードをフィルターできます。
    - Name では、ノード名で検索できます。
    - Label では、事前に定義されたラベルを選択して検索できます。
  - OpenShift Container Storage のラックラベルがデータセンターの物理ラックに合わせて調整されていることを確認し、障害ドメインのレベルで二重ノードに障害が発生しないようにします。

  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。

- **Storage Class** ドロップダウンリストから **localblock** を選択します。

f. **Create** をクリックします。

注記

Create ボタンは、最低でも 3 つのワーカーノードを選択した後にのみ有効になります。

デプロイメントに成功すると、3 つのストレージデバイスを持つストレージクラスターが作成されます。これらのデバイスは、選択したノードの 3 つに分散されます。この設定では、3 のレプリケーション係数が使用されます。初期クラスターをスケーリングするには、「ストレージノードのスケーリング」を参照してください。

検証手順

OpenShift Container Storage インストールの検証 について参照してください。

第2章 内部モードの OPENSHIFT 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。

2.1. 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2.1「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2.1 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	**ocs-operator-*** （任意のワーカーノードに 1 Pod）
Rook-ceph Operator	**rook-ceph-operator-*** （任意のワーカーノードに 1 Pod）
Multicloud Object Gateway	• **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) • **nooba-db-*** (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）
RGW	**rook-ceph-rgw-ocs-storagecluster-cephobjectstore-*** (ストレージノードに分散する 2 Pod)

CSI	• **cephfs**
	  ◦ **csi-cephfsplugin-*** (各ワーカーノードに 1 Pod)
	  ◦ **csi-cephfsplugin-provisioner-*** (ストレージノードに分散する 2 Pod)
	• **rbd**
	  ◦ **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-drain-canary	**rook-ceph-drain-canary-*** （各ストレージノードに 1 Pod）
rook-ceph-crashcollector	**rook-ceph-crashcollector-*** （各ストレージノードに 1 Pod）
OSD	• **rook-ceph-osd-*** (各デバイス用に 1 Pod) • **rook-ceph-osd-prepare-ocs-deviceset-*** (各デバイス用に 1 Pod)

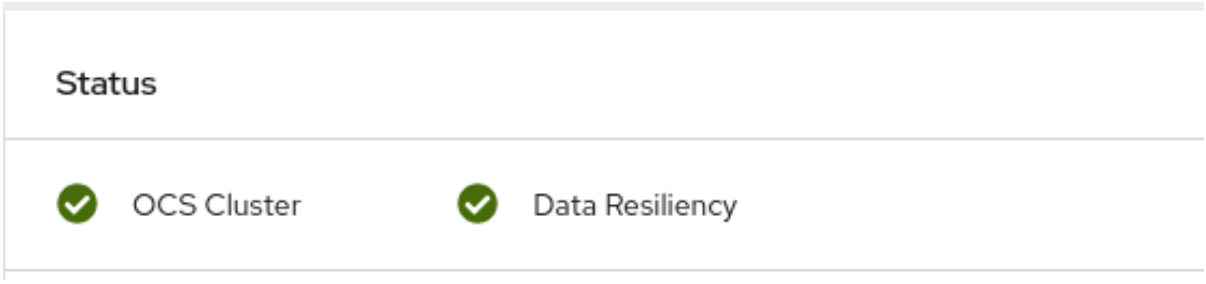
2.2. OPENSHIFT CONTAINER STORAGE クラスターが正常であることの確認

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を確認できます。詳細は、『OpenShift Container Storage のモニタリング』を参照してください。

- 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。

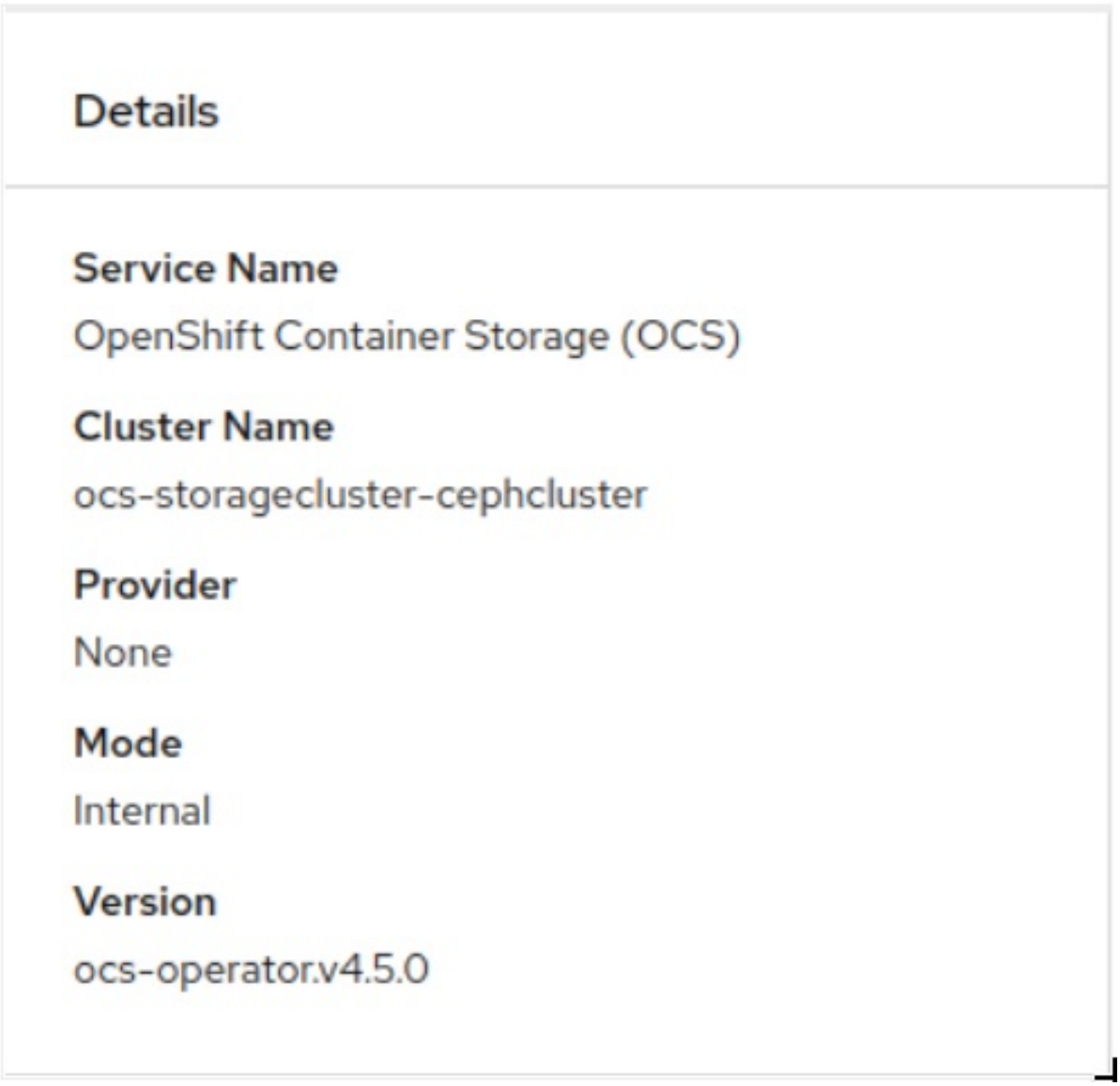
- **Status カード** で、以下の画像のように **OCS Cluster** に緑色のチェックマークが表示されていることを確認します。

図2.1 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適切に表示されていることを確認します。

図2.2 Persistent Storage Overview ダッシュボードの Details カード

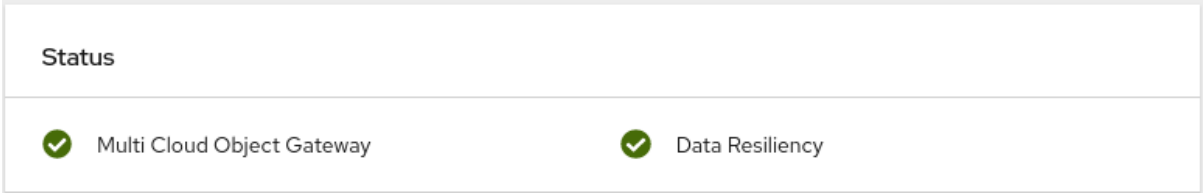


2.3. MULTICLOUD OBJECT GATEWAY が正常であることの確認

オブジェクトサービスダッシュボードを使用して、OpenShift Container Storage クラスターの正常性を確認できます。詳細は、『OpenShift Container Storage のモニタリング』を参照してください。

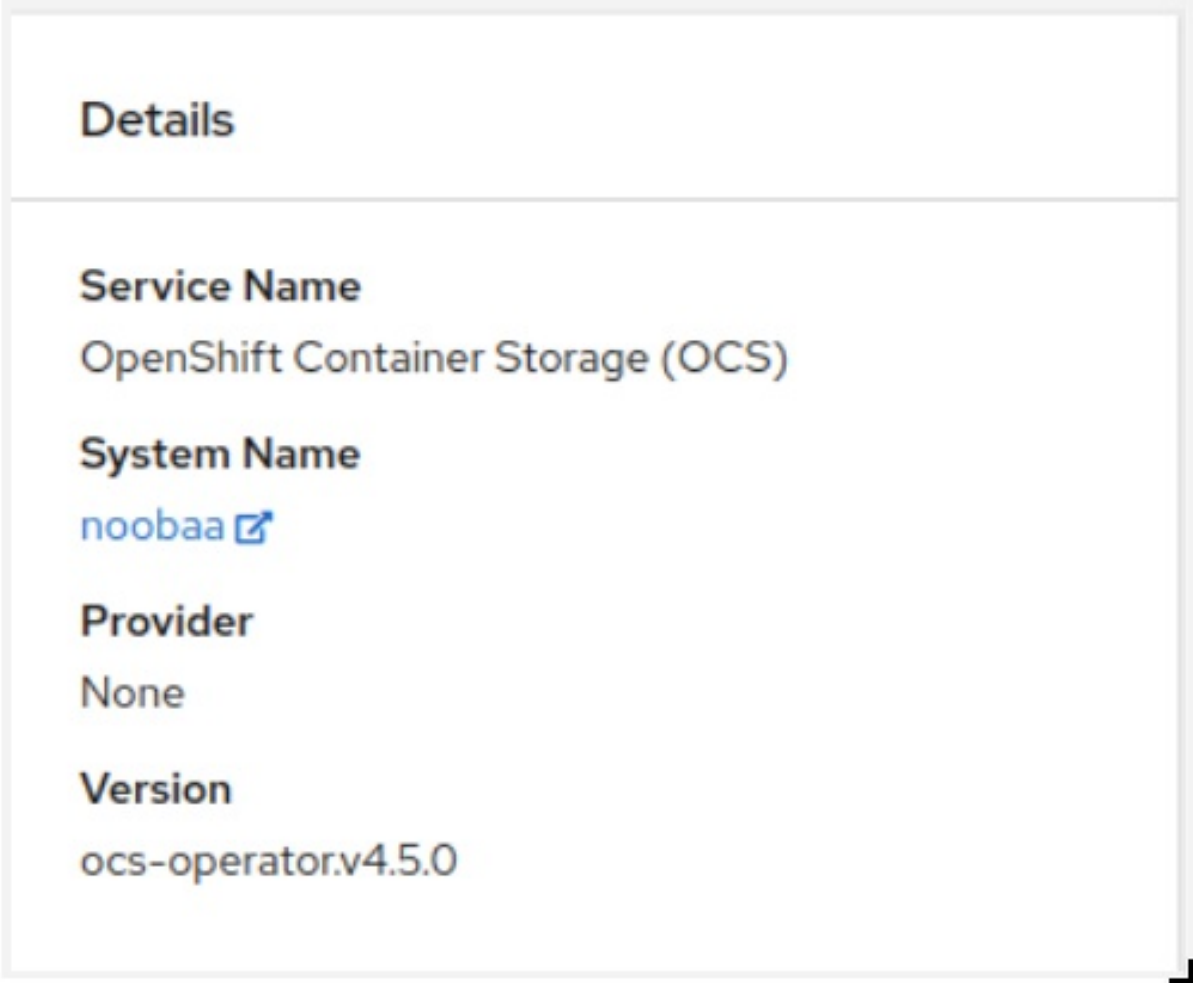
- OpenShift Web コンソールの左側のペインから **Home → Overview** をクリックし、**Object Service** タブをクリックします。
- **Status カード** で、以下のように Multicloud Object Gateway (MCG) ストレージに緑色のチェックマークが表示されていることを確認します。

  図2.3 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適切に表示されることを確認します。

  図2.4 Object Service Overview ダッシュボードの Details カード



2.4. 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**
- **ocs-storagecluster-ceph-rgw**

第3章 OPENSHIFT CONTAINER STORAGE のアンインストール

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

このセクションの手順を使用して、ユーザーインターフェースから Uninstall オプションを使用せずに OpenShift Container Storage をアンインストールします。

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。一部の Pod がリソースまたはノードの不足により正常に終了しないと、削除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 Persistent Volume Claim（永続ボリューム要求、PVC）または Object Bucket Claim（オブジェクトバケット要求）を使用していないことを確認します。PVC および OBC はアンインストールプロセスで削除されます。

手順

1. OpenShift Container Storage ベースのストレージクラスプロビジョナーを使用する PVC および OBC をクエリーします。
   以下は例になります。

   ```
   $ oc get pvc -o=jsonpath='{range .items[?(@.spec.storageClassName=="ocs-storagecluster-ceph-rbd")]}{"Name: "}{@.metadata.name}{" Namespace: "}{@.metadata.namespace}{" Labels: "}{@.metadata.labels}{"\n"}{end}' --all-namespaces|awk '! ( /Namespace: openshift-storage/ && /app:noobaa/ )' | grep -v noobaa-default-backing-store-noobaa-pvc
   ```

   ```
   $ oc get pvc -o=jsonpath='{range .items[?(@.spec.storageClassName=="ocs-storagecluster-cephfs")]}{"Name: "}{@.metadata.name}{" Namespace: "}{@.metadata.namespace}{"\n"}{end}' --all-namespaces
   ```

   ```
   $ oc get obc -o=jsonpath='{range .items[?(@.spec.storageClassName=="ocs-storagecluster-ceph-rgw")]}{"Name: "}{@.metadata.name}{" Namespace: "}{@.metadata.namespace}{"\n"}{end}' --all-namespaces
   ```

   ```
   $ oc get obc -o=jsonpath='{range .items[?(@.spec.storageClassName=="openshift-storage.noobaa.io")]}{"Name: "}{@.metadata.name}{" Namespace: "}{@.metadata.namespace}{"\n"}{end}' --all-namespaces
   ```

2. 以下の手順に従って、直前の手順に記載されている PVC および OBC が削除されていることを確認します。
   モニタリングスタック、クラスターロギング Operator、またはイメージレジストリーの設定の一部として PVC を作成した場合は、必要に応じて以下のセクションで説明されているクリーンアップ手順を実行する必要があります。

   - 「OpenShift Container Storage からのモニタリングスタックの削除」
   - 「OpenShift Container Storage からの OpenShift Container Platform レジストリーの削除」

- 「OpenShift Container Storage からのクラスターロギング Operator の削除」
残りの PVC または OBC のそれぞれに、以下の手順を実行します。

  a. PVC または OBC を使用する Pod を判別します。

  b. **Deployment**、**StatefulSet**、**DaemonSet**、**Job**、またはカスタムコントローラーなどの制御する側の API オブジェクトを特定します。
  各 API オブジェクトには、**OwnerReference** として知られるメタデータフィールドがあります。これは、関連付けられたオブジェクトの一覧です。**controller** フィールドが true に設定された **OwnerReference** は、**ReplicaSet**、**StatefulSet**、**DaemonSet** などの制御するオブジェクトを参照します。

  c. API オブジェクトが OpenShift Container Storage によって提供される PVC または OBC を使用していないことを確認します。オブジェクトを削除するか、ストレージを置き換える必要があります。プロジェクトオーナーに、オブジェクトを安全に削除または変更できることを確認するよう依頼します。

  注記

  noobaa Pod は無視できます。

  d. OBC を削除します。

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

  e. 作成したカスタムバケットクラスを削除します。

  ```
  $ oc get bucketclass -A  | grep -v noobaa-default-bucket-class
  ```

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

  f. カスタム Multi Cloud Gateway バッキングストアを作成している場合は、それらを削除します。

  - バッキングストアの一覧を表示し、これらをメモします。

    ```
    for bs in $(oc get backingstore -o name -n openshift-storage | grep -v noobaa-default-backing-store); do echo "Found backingstore $bs"; echo "Its has the following pods running :"; echo "$(oc get pods -o name -n openshift-storage | grep $(echo ${bs} | cut -f2 -d/))"; done
    ```

  - 上記の各バッキングストアを削除し、依存するリソースも削除されていることを確認します。

    ```
    for bs in $(oc get backingstore -o name -n openshift-storage | grep -v noobaa-default-backing-store); do echo "Deleting Backingstore $bs"; oc delete -n openshift-storage $bs; done
    ```

  - 上上記のバッキングストアのいずれかが pv-pool をベースとする場合、対応する Pod および PVC も削除してください。

    ```
    $ oc get pods -n openshift-storage | grep noobaa-pod | grep -v noobaa-default-backing-store-noobaa-pod
    ```

```
$ oc get pvc -n openshift-storage --no-headers | grep -v noobaa-db | grep
noobaa-pvc | grep -v noobaa-default-backing-store-noobaa-pvc
```

g. 手順 1 に記載されている残りの PVC を削除します。

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

3. バッキングローカルボリュームオブジェクトを一覧表示し、これをメモします。結果がない場合は、手順 7 と 8 に進みます。

```
$ for sc in $(oc get storageclass|grep 'kubernetes.io/no-provisioner' |grep -E $(oc get
storagecluster -n openshift-storage -o jsonpath='{
.items[*].spec.storageDeviceSets[*].dataPVCTemplate.spec.storageClassName}' | sed 's/
/|/g')| awk '{ print $1 }');
do
   echo -n "StorageClass: $sc ";
   oc get storageclass $sc -o jsonpath=" { 'LocalVolume: ' }{
.metadata.labels['local\.storage\.openshift\.io/owner-name'] } { '\n' }";
done
```

出力例:

```
StorageClass: localblock  LocalVolume: local-block
```

4. **StorageCluster** 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
```

5. 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、別のプロジェクトに切り替える必要があります。

a. openshift-storage がアクティブな namespace の場合に別の namespace に切り替えます。以下は例になります。

```
$ oc project default
```

b. openshift-storage namespace を削除します。

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

c. 約 5 分間待機し、プロジェクトが正常に削除されたかどうかを確認します。

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

出力:

```
Error from server (NotFound): namespaces "openshift-storage" not found
```

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。

6. 各ノードでストレージ Operator アーティファクトをクリーンアップします。

```
$ 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 rm -rfv /var/lib/rook; done
```

削除されたディレクトリー **/var/lib/rook** が出力に表示されることを確認します。

ディレクトリーが存在しないことを確認します。

```
$ 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
```

7. デプロイメント時に作成されたローカルボリュームを削除し、手順 3 に記載されている各ローカルボリュームについてこれを繰り返します。
ローカルボリュームごとに、以下を実行します。

   a. 変数 **LV** を LocalVolume の名前に設定し、変数 **SC** を手順 3 に一覧表示されている StorageClass の名前に設定します。
   以下に例を示します。

```
$ LV=local-block
```

```
$ SC=localblock
```

   b. 後にクリーンアップするデバイスを一覧表示し、これをメモします。

```
$ oc get localvolume -n local-storage $LV -o jsonpath='{
.spec.storageClassDevices[*].devicePaths[*] }'
```

   出力例:

```
/dev/disk/by-id/nvme-xxxxxx
/dev/disk/by-id/nvme-yyyyyy
/dev/disk/by-id/nvme-zzzzzz
```

   c. ローカルボリュームリソースを削除します。

```
$ oc delete localvolume -n 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 ...
```

8. 手順 3 に一覧表示されている各ローカルボリュームのディスクを消去して、それらを再利用できるようにします。

a. ストレージノードを一覧表示します。

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

出力例:

```
NAME     STATUS   ROLES    AGE     VERSION
node-xxx  Ready    worker   4h45m  v1.18.3+6c42de8
node-yyy  Ready    worker   4h46m  v1.18.3+6c42de8
node-zzz  Ready    worker   4h45m  v1.18.3+6c42de8
```

b. プロンプトが表示されたらノードコンソールを取得し、 **chroot /host** コマンドを実行します。

```
$ oc debug node/node-xxx
Starting pod/node-xxx-debug ...
To use host binaries, run `chroot /host`
Pod IP: w.x.y.z
If you don't see a command prompt, try pressing enter.
sh-4.2# chroot /host
```

c. 手順 7(ii)で収集されたディスクパスを引用符内の **DISKS** 変数に保存します。

```
sh-4.2# DISKS="/dev/disk/by-id/nvme-xxxxxx
/dev/disk/by-id/nvme-yyyyyy /dev/disk/by-id/nvme-zzzzzz"
```

d. すべてのディスクで **sgdisk --zap-all** を実行します。

```
sh-4.4# for disk in $DISKS; do sgdisk --zap-all $disk;done
```

出力例:

```
Problem opening /dev/disk/by-id/nvme-xxxxxx for reading! Error is 2.
The specified file does not exist!
Problem opening '' for writing! Program will now terminate.
Warning! MBR not overwritten! Error is 2!
Problem opening /dev/disk/by-id/nvme-yyyyy for reading! Error is 2.
The specified file does not exist!
Problem opening '' for writing! Program will now terminate.
Warning! MBR not overwritten! Error is 2!
Creating new GPT entries.
GPT data structures destroyed! You may now partition the disk using fdisk or
other utilities.
NOTE
Ignore file-not-found warnings as they refer to disks that are on other machines.
```

e. シェルを終了し、他のノードについて手順を繰り返します。

```
sh-4.4# exit
exit
sh-4.2# exit
exit

Removing debug pod ...
```

9. **openshift-storage.noobaa.io** ストレージクラスを削除します。

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

10. 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-
```

注記

label <label> not found のようなラベルが解除されているノードについて表示される警告は無視できます。

11. すべての PV が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 | egrep 'ocs-storagecluster-ceph-rbd|ocs-storagecluster-cephfs'
```

```
# oc delete pv <pv name>
```

12. **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
storageclusterinitializations.ocs.openshift.io storageclusters.ocs.openshift.io
cephclients.ceph.rook.io --wait=true --timeout=5m
```

13. 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いことを確認します。

3.2. OPENSHIFT CONTAINER STORAGE からのモニタリングスタックの削除

このセクションでは、モニタリングスタックを 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                                                     STATUS   VOLUME
CAPACITY   ACCESS MODES   STORAGECLASS                AGE
persistentvolumeclaim/my-alertmanager-claim-alertmanager-main-0   Bound    pvc-0d519c4f-
15a5-11ea-baa0-026d231574aa   40Gi       RWO           ocs-storagecluster-ceph-rbd   8d
persistentvolumeclaim/my-alertmanager-claim-alertmanager-main-1   Bound    pvc-
0d5a9825-15a5-11ea-baa0-026d231574aa   40Gi       RWO           ocs-storagecluster-ceph-
rbd   8d
persistentvolumeclaim/my-alertmanager-claim-alertmanager-main-2   Bound    pvc-
0d6413dc-15a5-11ea-baa0-026d231574aa   40Gi       RWO           ocs-storagecluster-ceph-
rbd   8d
persistentvolumeclaim/my-prometheus-claim-prometheus-k8s-0        Bound    pvc-0b7c19b0-
15a5-11ea-baa0-026d231574aa   40Gi       RWO           ocs-storagecluster-ceph-rbd   8d
persistentvolumeclaim/my-prometheus-claim-prometheus-k8s-1        Bound    pvc-0b8aed3f-
15a5-11ea-baa0-026d231574aa   40Gi       RWO           ocs-storagecluster-ceph-rbd   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
```

3.3. OPENSHIFT CONTAINER STORAGE からの OPENSHIFT 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は、「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:
  emptyDir: {}
.
.
.
```

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
```

3.4. OPENSHIFT 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 を使用するように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
```