



# Red Hat OpenShift Container Storage 4.3

## Red Hat OpenShift Container Storage のデプロ イ

環境のインストールおよび設定方法



# Red Hat OpenShift Container Storage 4.3 Red Hat OpenShift Container Storage のデプロイ

---

環境のインストールおよび設定方法

## 法律上の通知

Copyright © 2020 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<sup>®</sup> is the registered trademark of Linus Torvalds in the United States and other countries.

Java<sup>®</sup> is a registered trademark of Oracle and/or its affiliates.

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

MySQL<sup>®</sup> is a registered trademark of MySQL AB in the United States, the European Union and other countries.

Node.js<sup>®</sup> 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<sup>®</sup> 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

Red Hat OpenShift Container Storage 4.3 のインストール方法については、本書をお読みください。

---

## 目次

<b>第1章 RED HAT OPENSIFT CONTAINER STORAGE のデプロイ</b> .....	<b>3</b>
1.1. RED HAT OPENSIFT CONTAINER STORAGE の既存の OPENSIFT CONTAINER PLATFORM へのデプロイ	3
1.2. ローカルストレージデバイスを使用した OPENSIFT CONTAINER STORAGE のインストール	9
<b>第2章 OPENSIFT CONTAINER STORAGE インストールの確認</b> .....	<b>24</b>
2.1. POD が実行中の状態にあることの確認	24
2.2. OPENSIFT CONTAINER STORAGE クラスターが正常であることの確認	25
2.3. MULTICLOUD OBJECT GATEWAY が正常であることを確認します。	26
2.4. ストレージクラスが作成され、一覧表示されることを確認します。	27
<b>第3章 OPENSIFT CONTAINER PLATFORM のアンインストール</b> .....	<b>29</b>
3.1. OPENSIFT CONTAINER STORAGE からのモニタリングスタックの削除	32
3.2. OPENSIFT CONTAINER STORAGE からの OPENSIFT CONTAINER PLATFORM レジストリーの削除	36
3.3. OPENSIFT CONTAINER STORAGE からのクラスターロギング OPERATOR の削除	38



# 第1章 RED HAT OPENSIFT CONTAINER STORAGE のデプロイ

OpenShift Container Storage 4.3 では、インストールは既存の Red Hat OpenShift Container Platform (OCP) ワーカーノードでのみサポートされています。「[Red Hat OpenShift Container Storage の既存の OpenShift Container Platform へのデプロイ](#)」の手順に従って OpenShift Container Storage をデプロイします。



## 注記

OpenShift Container Storage をネットワークが制限された環境でインストールする場合、デフォルトでインターネット接続が OpenShift Container Platform で想定され、**chronyd** が **\*.rhel.pool.ntp.org** サーバーを使用するように設定されるため、カスタム Network Time Protocol (NTP) 設定をノードに適用する必要があります。詳細は、<https://access.redhat.com/solutions/4828941> および「[chrony タイムサービスの設定](#)」を参照してください。

## 1.1. RED HAT OPENSIFT CONTAINER STORAGE の既存の OPENSIFT CONTAINER PLATFORM へのデプロイ

デプロイメントプロセスは、以下の2つの主要な部分で構成されます。

1. 「[Operator Hub を使用した Red Hat OpenShift Container Storage Operator のインストール](#)」の手順に従って OpenShift Container Storage Operator をインストールします。
2. 「[OpenShift Container Storage サービスの作成](#)」の手順に従って OpenShift Container Storage サービスを作成します。

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 ベースのホストについては、「[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](#)」の手順に従って、基礎となるファイルシステムへのコンテナのアクセスを有効にする必要があります。

### 1.1.1. Operator Hub を使用した Red Hat OpenShift Container Storage Operator のインストール

Red Hat OpenShift Container Storage は、Amazon Web Services (AWS) および VMware vSphere プラットフォームで 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 (OCP) クラスターにログインする必要があります。
- OCP 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。
- 以下のように、**openshift-storage** という namespace を作成する必要があります。
  1. 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。
  2. **Create Namespace** をクリックします。

3. Create Namespace ダイアログボックスで、Name には **openshift-storage** を、Labels には **openshift.io/cluster-monitoring=true** を入力します。このラベルは、ダッシュボードを取得するために必要です。
4. **Default Network Policy** に **No restrictions** オプションを選択します。
5. **Create** をクリックします。



## 注記

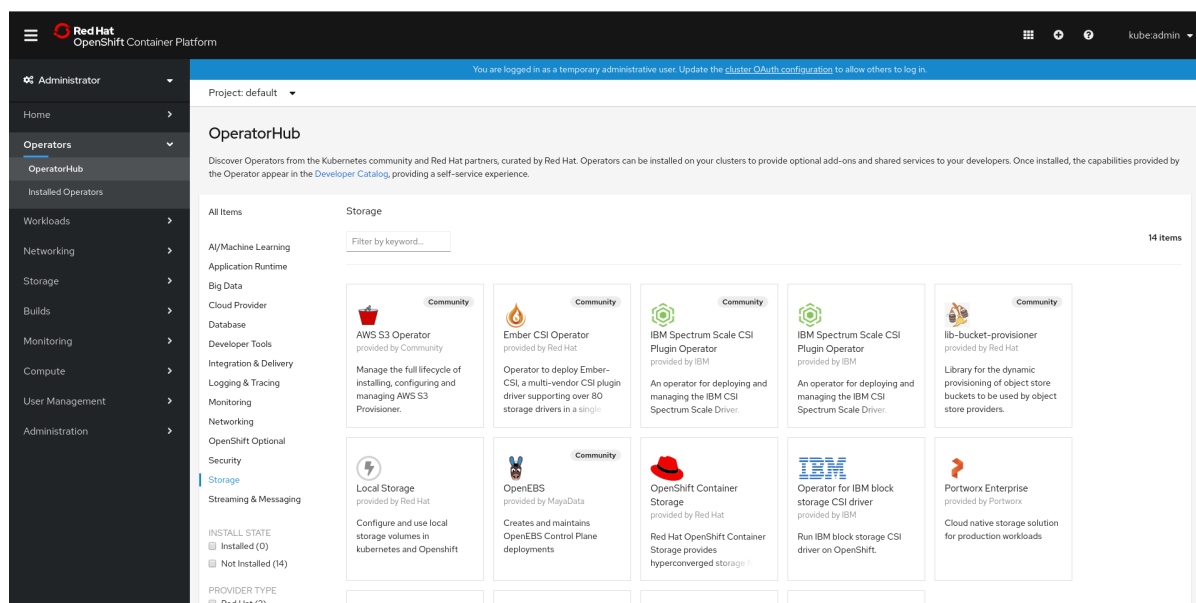
OpenShift Container Storage のクラスター全体でのデフォルトノードセクターを上書きする必要がある場合は、コマンドラインインターフェースで以下のコマンドを使用し、**openshift-storage** namespace の空のノードセクターを指定できます。

```
$ 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. Operator の一覧から **OpenShift Container Storage Operator** を検索し、これをクリックします。
3. OpenShift Container Storage Operator ページで、**Install** をクリックします。
4. Create Operator Subscription ページで、Installation Mode、Update Channel、および Approval Strategy オプションを選択できます。



## 図1.2 Create Operator Subscription ページ

OperatorHub > Operator Subscription

## Create Operator Subscription

Install your Operator by subscribing to one of the update channels to keep the Operator up to date. The strategy determines either manual or automatic updates.

**Installation Mode \***

All namespaces on the cluster (default)  
This mode is not supported by this Operator

A specific namespace on the cluster  
Operator will be available in a single namespace only.

**PR** openshift-storage

**Update Channel \***

stable-4.2

stable-4.3

**Approval Strategy \***

Automatic

Manual

**OpenShift Container Storage**  
provided by Red Hat, Inc

**Provided APIs**

**OCS [Internal] OCS Initialization**  
[This resource is not intended to be created or managed by users.] OCS Initialization represents the initial data to be created when the OCS operator is installed.

**SCI [Internal] StorageCluster Initialization**  
[This resource is not intended to be created or managed by users.] StorageCluster Initialization represents a set of tasks the OCS operator wants to implement for every StorageCluster it encounters.

**Subscribe** **Cancel**

a. Installation Mode オプションに **A specific namespace on the cluster** を選択します。

- ドロップダウンメニューから **openshift-storage** namespace を選択します。

b. 要件に基づいて **stable-4.3** を更新チャンネルとして選択します。

c. 承認ストラテジーを選択します。

- **Automatic** は、OpenShift Container Platform が OpenShift Container Storage を自動的にアップグレードすることを指定します。
- **Manual** は、OpenShift Container Storage を手動でアップグレードする際に制御できることを指定します。

5. **Subscribe** をクリックします。

## 図1.3 インストールされた Operator

Name	Namespace	Deployment	Status	Provided APIs
lib-bucket-provisioner 1.0.0 provided by Red Hat	openshift-storage	lib-bucket-provisioner	Succeeded Up to date	ObjectBucketClaim ObjectBucket
OpenShift Container Storage 4.3.0 provided by Red Hat, Inc	openshift-storage	ocs-operator	Succeeded Up to date	[Internal] OCS Initialization StorageCluster [Internal] StorageCluster Initialization [Internal] Ceph Cluster View 7 more...

Installed Operators ページには、Operator のステータスが表示されます。

- `lib-bucket-provisioner` および `OpenShift Container Storage Operator` がステータス **Succeeded** を表示することを確認します。

## 1.1.2. OpenShift Container Storage サービスの作成

OpenShift Container Storage Operator を Amazon Web Services (AWS) および VMware vSphere プラットフォームの両方のユーザーによってプロビジョニングされたクラウドにインストールした後に、新規の OpenShift Container Storage サービスを作成する必要があります。

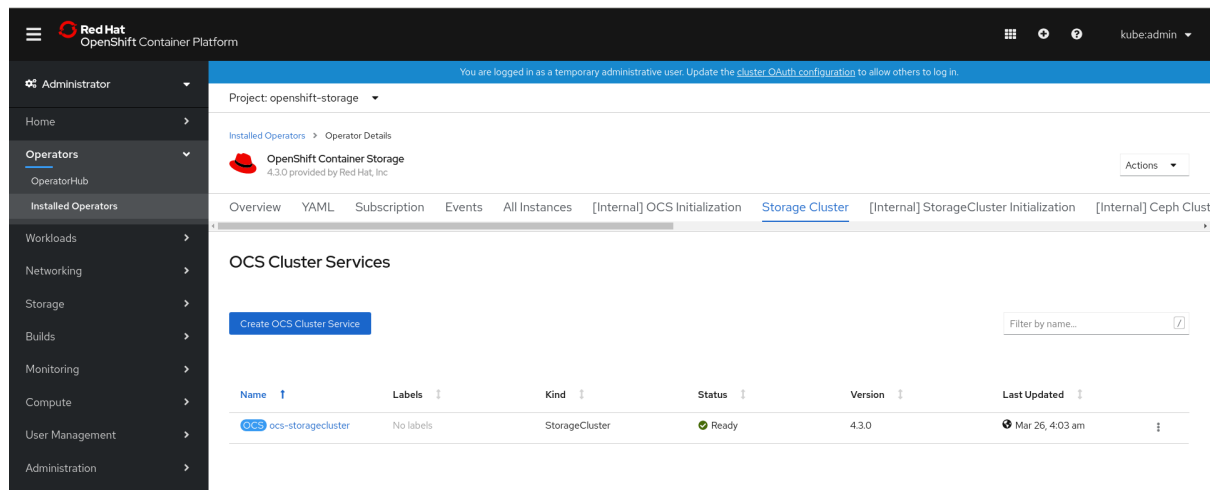
### 前提条件

- OpenShift Container Storage Operator は Operator Hub からインストールする必要があります。詳細は、「[Installing OpenShift Container Storage Operator using the Operator Hub](#)」を参照してください。

### 手順

1. OpenShift Web コンソールの左側のペインで **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。
2. Installed Operator ページで、**Project** ドロップダウンリストから **openshift-storage** を選択し、**openshift-storage** プロジェクトに切り替えます。
3. **OpenShift Container Storage Operator** をクリックします。  
OpenShift Container Storage Operator は **OCSInitialization** リソースを自動的に作成します。
4. OpenShift Container Storage Operator ページで右側にスクロールし、**Storage Cluster** タブをクリックします。

図1.4 OpenShift Container Storage Operator ページ



5. **OCS Cluster Services** ページで、**Create OCS Cluster Service** をクリックします。

## 図1.5 新規 OCS サービスページの作成

Project: openshift-storage ▼

## Create New OCS Service

OCS runs as a cloud-native service for optimal integration with applications in need of storage, and handles the scenes such as provisioning and management.

## Select Nodes


Selected nodes will be labeled with `cluster.ocs.openshift.io/openshift-storage=""` to create the OCS Service.
 A bucket will be created to provide the OCS Service.

Select at least 3 nodes in different failure domains you wish to use. \*

Filter by name... 

<input type="checkbox"/>	Name	Role	Location	CPU	Memory
<input type="checkbox"/>	 ip-10-0-139-2.us-east-2.compute.internal	worker	us-east-2a	16	60.89 GiB
<input type="checkbox"/>	 ip-10-0-158-163.us-east-2.compute.internal	worker	us-east-2b	16	61.54 GiB
<input type="checkbox"/>	 ip-10-0-140-101.us-east-2.compute.internal	worker	us-east-2a	16	60.89 GiB
<input type="checkbox"/>	 ip-10-0-171-124.us-east-2.compute.internal	worker	us-east-2c	16	60.89 GiB
<input type="checkbox"/>	 ip-10-0-152-33.us-east-2.compute.internal	worker	us-east-2b	16	61.54 GiB
<input type="checkbox"/>	 ip-10-0-169-14.us-east-2.compute.internal	worker	us-east-2c	16	61.54 GiB

0 node(s) selected

Storage Class 
 gp2 ▼
OCS Service Capacity 

2 TiB ▼

0.5 TiB  
SmallScale2 TiB  
Standard4 TiB  
LargeScale

## 6. Create New OCS Service ページで、以下を実行します。

- OpenShift Container Storage サービスを使用するために、利用可能なノードの一覧から3つ以上のワーカーノードを選択します。ノードが異なる **Location** にあることを確認します。
- ストレージクラス** は、AWS の場合はデフォルトで **gp2** に設定され、VMware の場合は **thin** になります。
- ドロップダウンリストから **OCS Service Capacity** を選択します。

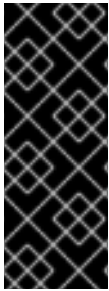


## 注記

ここで初期ストレージ容量を選択すると、この増分値でのみ容量を追加できます。

## 7. Create をクリックします。

**Create** ボタンは、3つのノードを選択した後にのみ有効になります。3つのボリュームからなる新規ストレージクラスターは、1ワーカーノードごとに1つのボリュームを設定して作成されます。デフォルト設定では、レプリケーション係数3を使用します。



### 重要

アプリケーション Pod は、OpenShift Container Storage ノードまたは OpenShift Container Storage 以外のノードのいずれかで作成し、アプリケーションを実行できます。ただし、テイントをノードに適用して、それらのノードに OpenShift Container Storage 専用を使用するようにマーク付け、アプリケーション Pod をそれらのノードで実行しないことが推奨されます。詳細は、<https://access.redhat.com/solutions/4827161> を参照してください。

### 検証手順

- OpenShift Container Storage 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。

### 1.1.3. 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) 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アクセスは自動的に提供されません。これは RHSTOR-787 で追跡されるバグです。



### 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や出力がない場合は、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
```

3. 必要なパッケージをインストールします。

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

4. 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
```

5. コンテナがこのノードでホストされる OpenShift Container Storage にアクセスできることを確認します。

## 1.2. ローカルストレージデバイスを使用した OPENSIFT CONTAINER STORAGE のインストール

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、Amazon EC2、および VMware 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。



### 重要

ローカルストレージ Operator を使用したベアメタル、Amazon EC2、および VMware 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。

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には、以下の手順を実行します。

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 OpenShift Container Storage Operator をインストール](#)します。
3. [ローカルストレージ Operator をインストール](#)します。
4. [利用可能なストレージデバイスを見つけ](#)ます。
5. 要件に基づいて OpenShift Container Storage クラスターを作成します。
  - Amazon EC2 の場合は、[Amazon EC2 での OpenShift Container Storage クラスターの作成](#) についての手順に従います。
  - VMware の場合は、[VMware での OpenShift Container Storage クラスターの作成](#) についての手順に従います。
  - ベアメタルの場合は、[ベアメタルでの OpenShift Container Storage クラスターの作成](#) についての手順に従います。

### 1.2.1. 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。
  - 各ワーカーノードには、最低 8 CPU および 64 GB のメモリーが必要です。
  - 3 つのワーカー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。

- 3つ以上のラベルが付けられたノードが必要です。
  - 各ワーカーノードには 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 <NodeName> cluster.ocs.openshift.io/openshift-storage="
```



### 注記

このラベルは、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すべてのワーカーノードに追加します。

- ローカルストレージ Operator 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他のストレージプロバイダーは存在しません。

## 1.2.2. 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 Amazon EC2、VMware、およびベアメタルインフラストラクチャーの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。

- 以下のように、**local-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1.6 Create Operator Subscription ページ

OperatorHub &gt; Operator Subscription

## Create Operator Subscription

Install your Operator by subscribing to one of the update channels to keep the Operator up to date. The strategy determines either manual or automatic updates.

## Installation Mode \*

- All namespaces on the cluster (default)  
Operator will be available in all namespaces.
- A specific namespace on the cluster  
Operator will be available in a single namespace only.

PR local-storage

## Update Channel \*

- 4.2
- 4.2-s390x
- 4.3

## Approval Strategy \*

- Automatic
- Manual



Local Storage

provided by Red Hat

## Provided APIs



Local Volume operator

Manage local storage volumes for OpenShift

Subscribe

Cancel

- Installation Mode オプションに **A specific namespace on the cluster** を選択します。
  - ドロップダウンメニューから **local-storage** namespace を選択します。
- Update Channel オプションの必要な値を選択します。
- 必要な Approval Strategy を選択します。
- Subscribe をクリックします。
- 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。

## 1.2.3. 利用可能なストレージデバイスの検索

この手順を使用して、ベアメタル、Amazon EC2、または VMware ストレージデバイスの PV を作成する前に、OpenShift Container Storage ラベル (**cluster.ocs.openshift.io/openshift-storage=**) を使用してラベルが付けられた 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

- 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
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
```

- 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的 **by-id** デバイス名を見つけます。

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

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 1.7T  0 disk
nvme1n1                             259:1   0 1.7T  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。

- 各デバイスのハードウェアシリアル番号に応じて、一意的 **by-id** デバイス名を見つけます。

```
sh-4.4# ls -l /dev/disk/by-id/
total 0
lrwxrwxrwx. 1 root root 10 Mar 17 16:24 dm-name-coreos-luks-root-nocrypt -> ../../dm-0
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
lrwxrwxrwx. 1 root root 13 Mar 17 16:24 nvme-nvme.1d0f-
4157533130333832453544373434313439344543-
416d617a6f6e20454332204e564d6520496e7374616e63652053746f72616765-00000001 ->
../../nvme0n1
lrwxrwxrwx. 1 root root 13 Mar 17 16:24 nvme-nvme.1d0f-
4157533630333832453544373434313439344543-
416d617a6f6e20454332204e564d6520496e7374616e63652053746f72616765-00000001 ->
../../nvme1n1
```

この例では、すべての OpenShift Container Storage ワーカーノードは Amazon EC2 タイプ **i3.2xlarge** になります。そのため、3つのワーカーノードすべてには同じタイプのマシンがありますが、**by-id** ID はすべてのローカルデバイスについて一意になります。**lsblk** コマンドは、それぞれのサイズが 1.7 TB の 2つのデバイス **nvme0n1** および **nvme1n1** を表示します。

OpenShift Container Storage ラベルが付けられた各ワーカーノード（最小 3つ）については、固有の **by-id** デバイス名を見つける必要があります。この例では、**by-id** デバイス名は以下のようになります。

- nvme-Amazon\_EC2\_NVMe\_Instance\_Storage\_AWS10382E5D7441494EC**
- nvme-Amazon\_EC2\_NVMe\_Instance\_Storage\_AWS60382E5D7441494EC**





## 注記

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について、デバイス名 **by-id** の検索を繰り返し実行する必要があります。詳細は、<https://access.redhat.com/solutions/4928841> を参照してください。

### 1.2.4. 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

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。これには、以下が含まれます。

1. **LocalVolume** CR の使用による PV の作成
2. 新規 **StorageClass** の作成

Amazon EC2 (ストレージ最適化: i3.2xlarge 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。



## 警告

OpenShift Container Storage の永続ストレージに Amazon EC2 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。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。

Amazon EC2 の一時ストレージを使用する場合は、以下が推奨されます。

- 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。
- OpenShift Container Storage は 8 CPU のあるインスタンスで利用可能な大半のリソースを使用するため、OpenShift Container Storage 専用を使用するノードにテイントのマークを付ける。

## 前提条件

- OpenShift Container Platform ワーカーノードに OpenShift Container Storage ラベルを付けます。このラベルは **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: local-storage
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-
        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 インスタンスには 2 つのディスクがあり、この例では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
```

- Pod が作成されているかどうかを確認します。

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

出力例:

NAME	READY	STATUS	RESTARTS	AGE
local-block-local-diskmaker-59rmn	1/1	Running	0	15m
local-block-local-diskmaker-6n7ct	1/1	Running	0	15m
local-block-local-diskmaker-jwtsn	1/1	Running	0	15m
local-block-local-provisioner-6ssxc	1/1	Running	0	15m
local-block-local-provisioner-swwvx	1/1	Running	0	15m
local-block-local-provisioner-zmv5j	1/1	Running	0	15m
local-storage-operator-7848bbd595-686dg	1/1	Running	0	15m

- 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1.7TB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	Bound	openshift-storage/ocs-
deviceset-2-0-2sf8c	localblock		14m		
local-pv-429d90ee	2328Gi	RWO	Delete	Bound	openshift-storage/ocs-
deviceset-1-1-nr95v	localblock		14m		
local-pv-4d0a62e3	2328Gi	RWO	Delete	Bound	openshift-storage/ocs-
deviceset-1-0-vnsf6	localblock		14m		
local-pv-55c05d76	2328Gi	RWO	Delete	Bound	openshift-storage/ocs-
deviceset-0-0-vtp84	localblock		14m		
local-pv-5c7b0990	2328Gi	RWO	Delete	Bound	openshift-storage/ocs-
deviceset-2-1-pkgfn	localblock		14m		
local-pv-a6b283b	2328Gi	RWO	Delete	Bound	openshift-storage/ocs-
deviceset-0-1-mz48g	localblock		14m		

- LocalVolume** CR の作成によって追加の **StorageClass** が作成されているかどうかを確認します。この **StorageClass** は、PVC を作成するために **StorageCluster** を作成する間に使用されます。

```
$ oc get sc
```

出力例:

NAME	PROVISIONER	AGE
gp2 (default)	kubernetes.io/aws-efs	7h14m
localblock	kubernetes.io/no-provisioner	7m46s

6. **localblock StorageClass** および作成される 3 つの PV を使用する **StorageCluster** CR を作成します。

**gp2** and **localblock** ストレージクラスを使用した **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
      requests:
        cpu: 1
    noobaa-core:
      limits:
        cpu: 2
        memory: 8Gi
      requests:
        cpu: 1
        memory: 8Gi
    noobaa-db:
      limits:
        cpu: 2
        memory: 8Gi
      requests:
        cpu: 1
        memory: 8Gi
  monPVCTemplate:
    spec:
      accessModes:
        - ReadWriteOnce
      resources:
        requests:
          storage: 10Gi
      storageClassName: gp2
      volumeMode: Filesystem
  storageDeviceSets:
    - count: 2
      dataPVCTemplate:
        spec:
          accessModes:
            - ReadWriteOnce
          resources:
            requests:
              storage: 2328Gi
          storageClassName: localblock
          volumeMode: Block
      name: ocs-deviceset
      placement: {}
```

```
portable: false
replica: 3
resources: {}
```



### 重要

**storageDeviceSets** のストレージサイズは raw ブロックデバイスのサイズ以下である必要があります。この要件を満たすには、値を 1 に設定できます。

## 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

「[Verifying your OpenShift Container Storage installation](#)」を参照してください。

## 1.2.5. VMware での OpenShift Container Storage クラスターの作成

以下の手順を使用して、VMware インフラストラクチャーにストレージクラスターを作成します。VMware は、以下の 3 つのタイプのローカルストレージをサポートします。

- 仮想マシンディスク (VMDK)
- raw デバイスマッピング (RDM)
- VMDirectPath I/O

## 前提条件

- VMware でローカルストレージデバイスを使用するために、各ノードに同じストレージタイプが割り当てられた 3 つのワーカーノードが必要です。
- **Filesystem** ボリュームモードの追加の StorageClass を作成し、**LocalVolume** CR を更新して適切なデバイス名を使用する必要があります。
- **Filesystem LocalVolume** CR を作成するには、OpenShift Container Storage ラベルおよび **StorageCluster** に使用されるストレージデバイスと共に、10GB VMDK を作成し、これを OpenShift ワーカーノードのそれぞれに割り当てる必要があります。
- OpenShift Container Platform ワーカーノードに OpenShift Container Storage ラベルを付けます。このラベルは **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'}
```

[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。

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. **LocalVolume** 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 ラベルをノードセレクターとして使用する **LocalVolume** CR の **local-storage-file.yaml** ファイルの例:

```
apiVersion: local.storage.openshift.io/v1
kind: LocalVolume
metadata:
  name: local-file
  namespace: local-storage
spec:
  nodeSelector:
    nodeSelectorTerms:
      - matchExpressions:
          - key: cluster.ocs.openshift.io/openshift-storage
            operator: In
            values:
              - ""
  storageClassDevices:
    - storageClassName: localfile
      volumeMode: Filesystem
      devicePaths:
        - /dev/disk/by-id/scsi-36000c29520486e45a6896c58a10de97d # <-- modify this line
        - /dev/disk/by-id/scsi-36000c29194bc3fd865a67057a29ec # <-- modify this line
        - /dev/disk/by-id/scsi-36000c2991133c4f1d6a604f3bc086967 # <-- modify this line
```



### 注記

OpenShift Container Platform デプロイメントに VMDK を作成し、これを OpenShift Container Platform ノードに割り当てるために利用できる **thin StorageClass** がある場合、**thin** を **localfile** の代わりに使用できます。この例のように **Filesystem volumeMode** で **LocalVolume** CR を使用して **localfile StorageClass** を作成する必要はありません。

2. ファイルシステム PV の **LocalVolume** CR を作成します。

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

3. 新しい **localfile StorageClass** を確認します。

```
$ oc get sc | grep localfile
```

出力例:

NAME	PROVISIONER	AGE
localfile	kubernetes.io/no-provisioner	7m46s

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

OpenShift Storage Container ラベルをノードセクターとして使用する **LocalVolume** CR の **local-storage-block.yaml** の例:

```
apiVersion: local.storage.openshift.io/v1
kind: LocalVolume
metadata:
  name: local-block
  namespace: local-storage
spec:
  nodeSelector:
    nodeSelectorTerms:
      - matchExpressions:
          - key: cluster.ocs.openshift.io/openshift-storage
            operator: In
            values:
              - ""
  storageClassDevices:
    - storageClassName: localblock
      volumeMode: Block
      devicePaths:
        - /dev/disk/by-id/scsi-36000c2991c27c2e5ba7c47d1e4352de2 # <-- modify this line
        - /dev/disk/by-id/scsi-36000c29682ca9e347926406711f3dc4e # <-- modify this line
        - /dev/disk/by-id/scsi-36000c296aaf03a9b1e4b01d086bc6348 # <-- modify this line
```

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

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

6. Pod が作成されているかどうかを確認します。

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

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

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

出力例:

NAME	PROVISIONER	AGE
localblock	kubernetes.io/no-provisioner	8m38s

8. **Available** のステータスで作成されている PV を確認します。

```
$ oc get pv
```

出力例:

NAME	CAPACITY	ACCESS MODES	RECLAIM POLICY	STATUS	CLAIM
local-pv-150fdc87	100Gi	RWO	Delete	Available	localblock
2m11s					
local-pv-183bfc0a	100Gi	RWO	Delete	Available	localblock
2m11s					

local-pv-b2f5cb25 2m21s	100Gi	RWO	Delete	Available	localblock
local-pv-ff902790 4m19s	10Gi	RWO	Delete	Available	localfile
local-pv-ea87e127 4m20s	10Gi	RWO	Delete	Available	localfile
local-pv-56fa56a1 4m30s	10Gi	RWO	Delete	Available	localfile

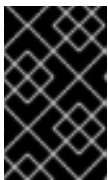
この例では、3つのボリュームがモニターストレージ (10 GB) に使用され、3つが OSD ストレージ (100 GB) に使用されます。

9. **localfile** および **localblock StorageClass** を使用する **StorageCluster** CR を作成します。**localfile** および **localblock** ストレージクラスを使用する **StorageCluster** CR **ocs-cluster-service-VMware.yaml** の例

```

apiVersion: ocs.openshift.io/v1
kind: StorageCluster
metadata:
  name: ocs-storagecluster
  namespace: openshift-storage
spec:
  manageNodes: false
  monPVCTemplate:
    spec:
      accessModes:
        - ReadWriteOnce
      resources:
        requests:
          storage: 10Gi
      storageClassName: localfile
      volumeMode: Filesystem
  storageDeviceSets:
    - count: 1
      dataPVCTemplate:
        spec:
          accessModes:
            - ReadWriteOnce
          resources:
            requests:
              storage: 2Ti
          storageClassName: localblock
          volumeMode: Block
      name: ocs-device-set
      placement: {}
      portable: false
      replica: 3
      resources: {}

```



### 重要

**monPVCTemplate**: および **storageDeviceSets** のストレージサイズは raw ブロックデバイスのサイズ以下である必要があります。この要件を満たすには、両方に 1 の値を設定できます。





## 注記

**thin StorageClass** は、利用可能な場合に **localfile** の代わりにモニターストレージを作成するために使用できます。

10. **StorageCluster** CR を作成します。

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

出力例:

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

## 検証手順

「[Verifying your OpenShift Container Storage installation](#)」を参照してください。

## 1.2.6. ベアメタルでの OpenShift Container Storage クラスターの作成

### 前提条件

- 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 Container Storage ラベルを付けます。このラベルは **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'}
```

[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。

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
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
  - /dev/disk/by-id/nvme-INTEL_SSDPEKKA128G7_BTPY0A60CB81128A # <-- modify
this line
  - /dev/disk/by-id/nvme-INTEL_SSDPEKKA128G7_BTPY0093D45E128A # <-- modify
this line
  - /dev/disk/by-id/nvme-INTEL_SSDPEKKA128G7_BTPYE46F6060128A # <-- 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。

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

4. PV が作成されているかどうかを確認します。

```
$ oc get pv
```

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

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

出力例:

```

NAME          PROVISIONER          AGE
localblock    kubernetes.io/no-provisioner    10m20s

```

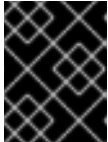
6. **monDataDirHostPath** および **localblock** ストレージクラスを使用して **StorageCluster** CR **cluster-service-metal.yaml** を作成します。

```

apiVersion: ocs.openshift.io/v1
kind: StorageCluster
metadata:
  name: ocs-storagecluster
  namespace: openshift-storage
spec:
  manageNodes: false
  monDataDirHostPath: /var/lib/rook
  storageDeviceSets:
  - count: 1
    dataPVCTemplate:
      spec:

```

```
accessModes:
- ReadWriteOnce
resources:
  requests:
    storage: 2Ti
storageClassName: localblock
volumeMode: Block
name: ocs-deviceset
placement: {}
portable: false
replica: 3
resources: {}
```



### 重要

**storageDeviceSets** のストレージサイズは raw ブロックデバイスのサイズ以下である必要があります。この要件を満たすには、値を 1 に設定できます。

7. **StorageCluster** CR を作成します。

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

出力例:

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

### 検証手順

「[Verifying your OpenShift Container Storage installation](#)」を参照してください。

## 第2章 OPENSIFT CONTAINER STORAGE インストールの確認

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 が実行中の状態にあることの確認

- 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。
- **Project** ドロップダウンリストから **openshift-storage** を選択します。  
Pod の数は OpenShift Container Platform にデプロイされるワーカーノードの数によって異なります。

#### 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手順を実行できます。

1. **openshift-storage** namespace の空のノードセレクターを指定します。

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

2. **DaemonSets** によって生成される元の Pod を削除します。

```
oc delete pod -l app=csi-cephfsplugin -n openshift-storage
oc delete pod -l app=csi-rbdplugin -n openshift-storage
```

**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2.13 つのワーカーノードからなるクラスターのストレージコンポーネントに対応する Pod

コンポーネント	Pod の数	Pod の名前
OpenShift Container Storage Operator	1	<b>ocs-operator-*</b>
Rook-ceph Operator	1	<b>rook-ceph-operator-*</b>
NooBaa	4	<ul style="list-style-type: none"> <li>● <b>noobaa-operator-*</b></li> <li>● <b>noobaa-core-*</b></li> <li>● <b>nooba-db-*</b></li> <li>● <b>noobaa-endpoint-*</b></li> </ul>

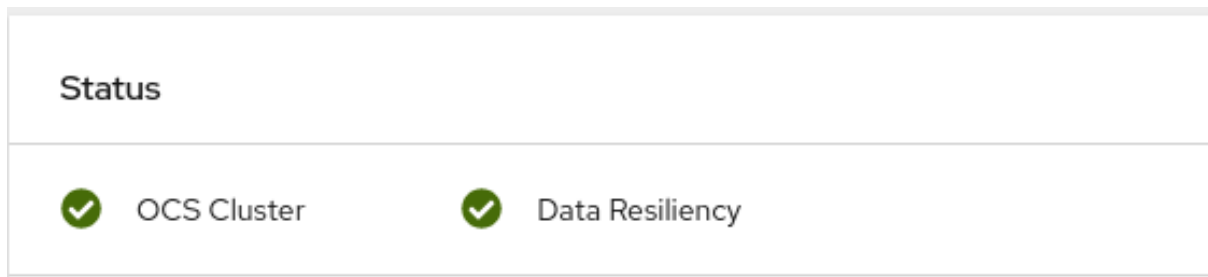
コンポーネント	Pod の数	Pod の名前
Mon	3	<ul style="list-style-type: none"> <li>● <b>rook-ceph-mon-a-*</b></li> <li>● <b>rook-ceph-mon-b-*</b></li> <li>● <b>rook-ceph-mon-c-*</b> (異なるノード上)</li> </ul>
CSI	10	<ul style="list-style-type: none"> <li>● <b>cephfs</b> (5 Pod) <ul style="list-style-type: none"> <li>○ <b>CSI-cephfsplugin-*</b> (異なるノードにある 3 Pod)</li> <li>○ <b>csi-cephfsplugin-provisioner-*</b> (2 Pod)</li> </ul> </li> <li>● <b>rbd</b> (5 Pod) <ul style="list-style-type: none"> <li>○ <b>CSI-rbdplugin-*</b> (異なるノードにある 3 Pod)</li> <li>○ <b>csi-rbdplugin-provisioner-*</b> (2 Pod)</li> </ul> </li> </ul>
OSD	6	<ul style="list-style-type: none"> <li>● <b>rook-ceph-osd-*</b> (異なるノードにある 3 Pod)</li> <li>● <b>rook-ceph-osd-prepare-ocs-deviceset-*</b> (3 Pod)</li> </ul>
rook-ceph-mgr	1	<b>rook-ceph-mgr-*</b>
mds	2	<b>rook-ceph-mds-ocs-storagecluster-cephfilesystem-*</b> (異なるノードにある 2 Pod)
rook-ceph-drain-canary	3	<b>rook-ceph-drain-canary-*</b> (3 pods)
rook-ceph-crashcollector	3	<b>rook-ceph-crashcollector-*</b> (3 pods)
lib-bucket-provisioner	1	<b>lib-bucket-provisioner-*</b>

## 2.2. OPENSIFT CONTAINER STORAGE クラスタ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。

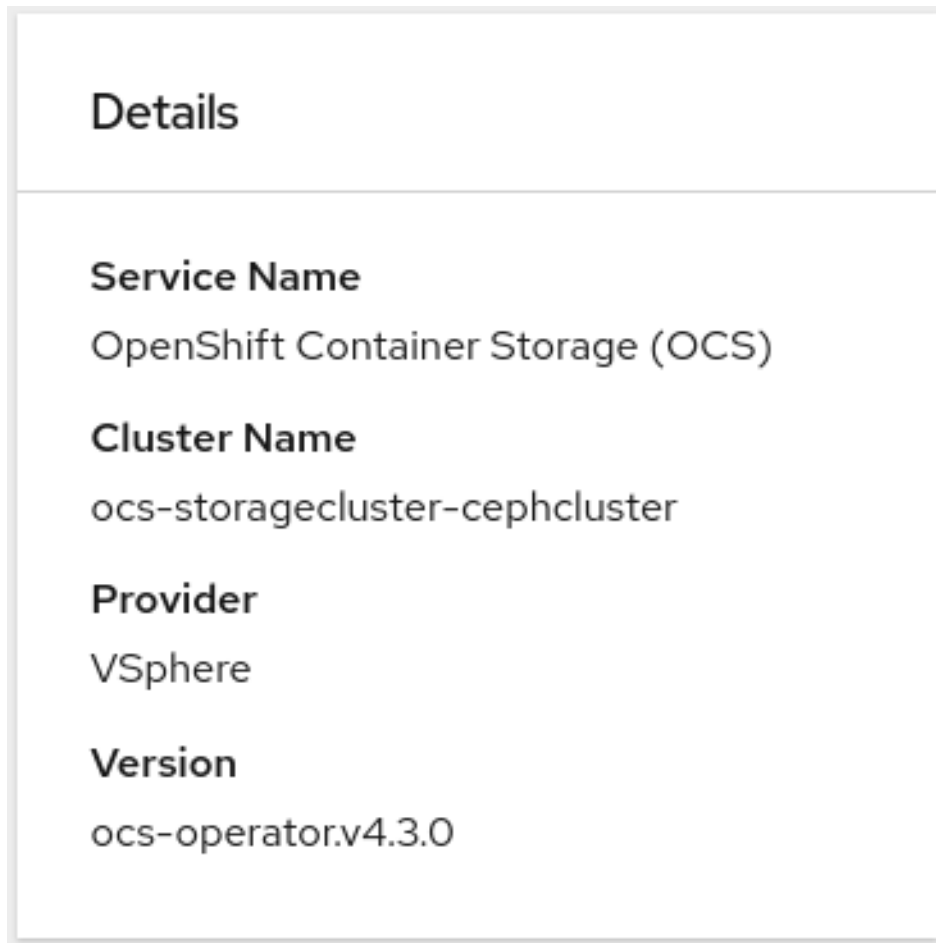
- OpenShift Web コンソールの左側のペインで **Home → Dashboards** をクリックし、**OCS PV** タブをクリックします。  
**Status** カードで、以下の画像のように **OCS Cluster** に緑色のチェックマークが表示されていることを確認します。

図2.1 Persistent Storage (OCS PV) 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 (OCS PV) ダッシュボードの Details カード

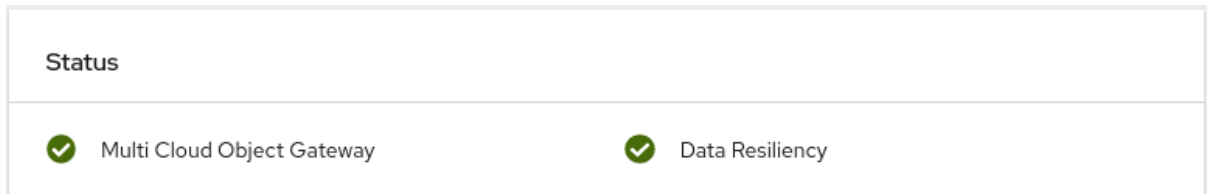


## 2.3. MULTICLOUD OBJECT GATEWAY が正常であることを確認します。

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。

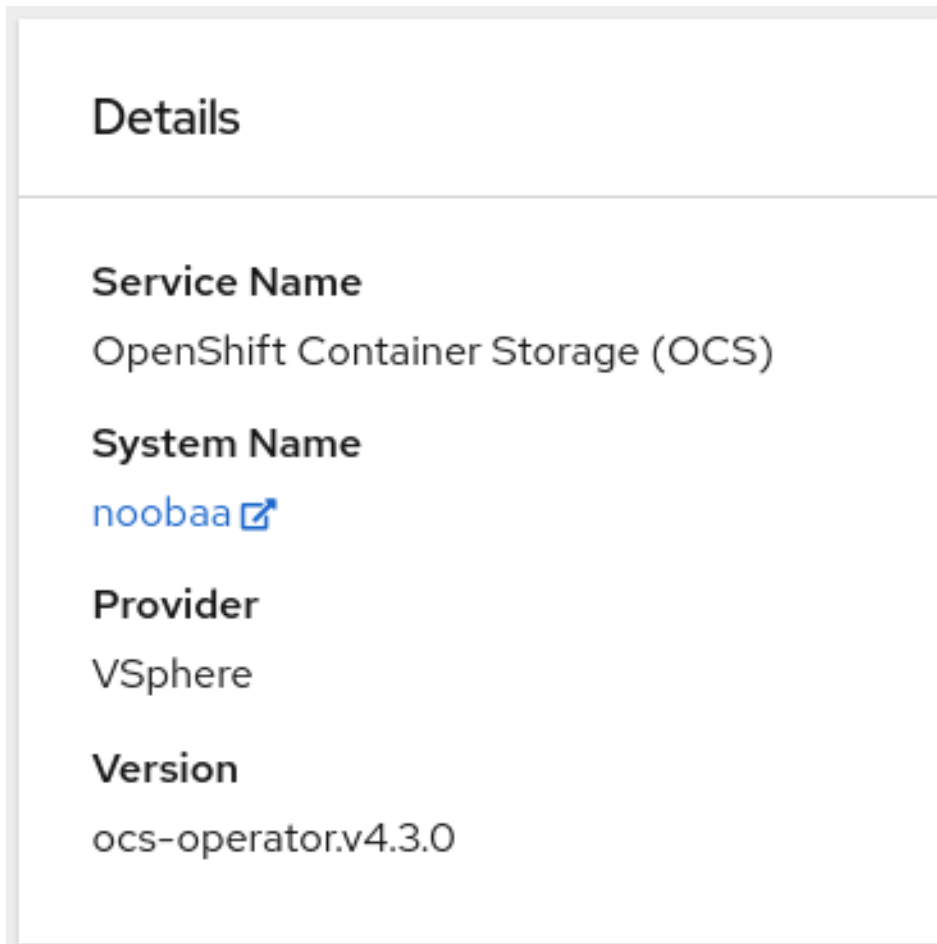
- OpenShift Web コンソールの左側のペインから **Home → Dashboards** をクリックし、**OCS Object Service** タブをクリックします。  
**Status** カードで、以下のように Multicloud Object Gateway (MCG) ストレージに緑色のチェックマークが表示されていることを確認します。

図2.3 Object Service 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 ダッシュボードの Details カード



## 2.4. ストレージクラスが作成され、一覧表示されることを確認します。

- 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3つ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**

You are logged in as a temporary administrative user. Update the [cluster OAuth configuration](#) to allow others to log in.

## Storage Classes

Create Storage Class

Filter by name...

Name ↑	Provisioner ↓	Reclaim Policy ↓
 ocs-storagecluster-ceph-rbd	openshift-storage.rbd.csi.ceph.com	Delete ⋮
 ocs-storagecluster-cephfs	openshift-storage.cephfs.csi.ceph.com	Delete ⋮
 openshift-storage.noobaa.io	openshift-storage.noobaa.io/obc	Delete ⋮
 thin - Default	kubernetes.io/vsphere-volume	Delete ⋮



## 第3章 OPENSIFT CONTAINER PLATFORM 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。
- OpenShift Container Storage ストレージクラスに基づいて 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、OBC) を使用するアプリケーションを削除してから、OpenShift Container Storage ストレージクラスを使用している PVC および OBC を削除します。

### 手順

1. ストレージクラスを一覧表示し、以下のストレージクラスプロビジョナーのストレージクラスをメモします。

- **openshift-storage.rbd.csi.ceph.com**
- **openshift-storage.cephfs.csi.ceph.com**
- **openshift-storage.noobaa.io/obc**

以下は例になります。

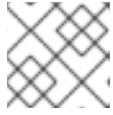
```
$ oc get storageclasses
NAME                                PROVISIONER                                AGE
gp2 (default)                       kubernetes.io/aws-ebs                      23h
ocs-storagecluster-ceph-rbd          openshift-storage.rbd.csi.ceph.com         22h
ocs-storagecluster-cephfs           openshift-storage.cephfs.csi.ceph.com     22h
openshift-storage.noobaa.io          openshift-storage.noobaa.io/obc           22h
```

2. 直前の手順に記載されているストレージクラスプロビジョナーを使用している PVC および OBC をクエリーします。

```
$ 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/ )'
```

```
$ 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=="openshift-storage.noobaa.io")]}{"Name: "}{@.metadata.name}{" Namespace: "}{@.metadata.namespace}{"\n"}{end}' --all-namespaces
```



## 注記

**openshift-storage** namespace の NooBaa PVC を無視します。

3. 以下の手順に従って、直前の手順に記載されている PVC が削除されていることを確認します。
  - a. PVC 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オブジェクトを特定します。  
各オブジェクトには、**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** などの制御する側のオブジェクトをポイントします。
  - c. プロジェクトの所有者を確認してからこれを削除し、オブジェクトを安全に削除できるようにします。
  - d. PVC を削除します。  
モニタリングスタック、クラスターロギング Operator、または prometheus 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」](#)
4. **StorageCluster** オブジェクトを削除します。

```
$ 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。

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

6. 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=
NAME                                STATUS  ROLES  AGE  VERSION
ip-10-0-134-54.us-east-2.compute.internal  Ready  worker  26m  v1.14.6+8e46c0036
ip-10-0-149-9.us-east-2.compute.internal   Ready  worker  26m  v1.14.6+8e46c0036
ip-10-0-170-32.us-east-2.compute.internal  Ready  worker  26m  v1.14.6+8e46c0036
```

7. それぞれのノードについて、以下を実行します。

- a. **oc debug node/<NODENAME>** コマンドを使用してノードコンソールを取得し、プロンプトが表示されたら **chroot /host** コマンドを実行します。

```
$ oc debug node/ip-10-0-134-54.us-east-2.compute.internal
Starting pod/ip-10-0-134-54us-east-2computeinternal-debug ...
To use host binaries, run `chroot /host`
```

```
Pod IP: 10.0.134.54
If you don't see a command prompt, try pressing enter.
sh-4.2# chroot /host
```

- b. OpenShift Container Storage コンポーネントをクリーンアップします。

```
sh-4.4# rm -rf /var/lib/rook
```

- c. **exit** を 2 回入力するか、または **Ctrl-D** を 2 回押してコンソールを終了します。

```
sh-4.4# exit
sh-4.2# exit
Removing debug pod ..
```

8. 手順 1 で一覧表示されている **openshift-storage** プロビジョナーの設定されているストレージクラスを削除します。

```
$ oc delete storageclass <storageclass-name> --wait=true --timeout=5m
```

以下は例になります。

```
$ oc delete storageclass ocs-storagecluster-ceph-rbd ocs-storagecluster-cephfs openshift-storage.noobaa.io --wait=true --timeout=5m
```

9. ストレージノードからテイントを削除します。

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



#### 注記

**taint "node.ocs.openshift.io/storage:" not found** のようなテイントが付けられていないノードについて表示される警告は無視できます。

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. **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 --wait=true --
timeout=5m
```

- OpenShift Container Storage がアンインストールされていることを確認するには、`openshift-storage` namespace がすでに存在しておらず、ストレージダッシュボードが UI に表示されないことを確認します。



## 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、<https://access.redhat.com/solutions/3881901> の記事を参照して 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。**Cephcluster**、**StorageCluster**、**NooBaa**、および **PVC** などの OpenShift オブジェクトにファイナライザーがある場合、それは namespace が **Terminating** の状態であることが原因である可能性があります。PVC にファイナライザーがある場合、関連付けられた Pod を削除してファイナライザーを削除します。

## 3.1. OPENSIFT 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せん。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

- 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	STORAGECLASS	STATUS	VOLUME	AGE
persistentvolumeclaim/my-alertmanager-claim-alertmanager-main-0	40Gi	RWO	ocs-storagecluster-ceph-rbd	Bound	pvc-0d519c4f-15a5-11ea-baa0-026d231574aa	8d
persistentvolumeclaim/my-alertmanager-claim-alertmanager-main-1	40Gi	RWO	ocs-storagecluster-ceph-rbd	Bound	pvc-0d5a9825-15a5-11ea-baa0-026d231574aa	8d
persistentvolumeclaim/my-alertmanager-claim-alertmanager-main-2	40Gi	RWO	ocs-storagecluster-ceph-rbd	Bound	pvc-0d6413dc-15a5-11ea-baa0-026d231574aa	8d
persistentvolumeclaim/my-prometheus-claim-prometheus-k8s-0	40Gi	RWO	ocs-storagecluster-ceph-rbd	Bound	pvc-0b7c19b0-15a5-11ea-baa0-026d231574aa	8d
persistentvolumeclaim/my-prometheus-claim-prometheus-k8s-1	40Gi	RWO	ocs-storagecluster-ceph-rbd	Bound	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。

Before editing

```
.  
. .  
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  
.  
.  
.
```

After editing

```

.
.
.
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. PVC を使用する Pod を一覧表示します。

この例では、PVC を使用する **alertmanagerMain** および **prometheusK8s** Pod は **Terminating** 状態にあります。PVC は完全に終了しないと削除できません。

```

$ oc get pod,pvc -n openshift-monitoring
NAME                                READY STATUS   RESTARTS AGE
pod/alertmanager-main-0             3/3 Terminating 0    10h
pod/alertmanager-main-1             3/3 Terminating 0    10h
pod/alertmanager-main-2             3/3 Terminating 0    10h
pod/cluster-monitoring-operator-84cd9df668-zhjfn 1/1 Running    0    18h
pod/grafana-5db6fd97f8-pmtbf        2/2 Running    0    10h
pod/kube-state-metrics-895899678-z2r9q 3/3 Running    0    10h
pod/node-exporter-4njxv             2/2 Running    0    18h
pod/node-exporter-b8ckz             2/2 Running    0    11h
pod/node-exporter-c2vp5             2/2 Running    0    18h
pod/node-exporter-cq65n             2/2 Running    0    18h
pod/node-exporter-f5sm7             2/2 Running    0    11h
pod/node-exporter-f852c             2/2 Running    0    18h
pod/node-exporter-l9zn7             2/2 Running    0    11h
pod/node-exporter-ngbs8             2/2 Running    0    18h
pod/node-exporter-rv4v9             2/2 Running    0    18h
pod/openshift-state-metrics-77d5f699d8-69q5x 3/3 Running    0    10h
pod/prometheus-adapter-765465b56-4tbxx 1/1 Running    0    10h
pod/prometheus-adapter-765465b56-s2qg2 1/1 Running    0    10h
pod/prometheus-k8s-0                6/6 Terminating 1    9m47s
pod/prometheus-k8s-1                6/6 Terminating 1    9m47s
pod/prometheus-operator-cbfd89f9-ldnwc 1/1 Running    0    43m
pod/telemeter-client-7b5ddb4489-2xfpz 3/3 Running    0    10h

NAME                                STATUS VOLUME
CAPACITY ACCESS MODES STORAGECLASS AGE
persistentvolumeclaim/ocs-alertmanager-claim-alertmanager-main-0 Bound pvc-

```

```

2eb79797-1fed-11ea-93e1-0a88476a6a64 40Gi RWO ocs-storagecluster-ceph-
rbd 19h
persistentvolumeclaim/ocs-alertmanager-claim-alertmanager-main-1 Bound pvc-
2ebeee54-1fed-11ea-93e1-0a88476a6a64 40Gi RWO ocs-storagecluster-ceph-
rbd 19h
persistentvolumeclaim/ocs-alertmanager-claim-alertmanager-main-2 Bound pvc-2ec6a9cf-
1fed-11ea-93e1-0a88476a6a64 40Gi RWO ocs-storagecluster-ceph-rbd 19h
persistentvolumeclaim/ocs-prometheus-claim-prometheus-k8s-0 Bound pvc-3162a80c-
1fed-11ea-93e1-0a88476a6a64 40Gi RWO ocs-storagecluster-ceph-rbd 19h
persistentvolumeclaim/ocs-prometheus-claim-prometheus-k8s-1 Bound pvc-
316e99e2-1fed-11ea-93e1-0a88476a6a64 40Gi RWO ocs-storagecluster-ceph-
rbd 19h

```

5. 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-name> --wait=true --timeout=5m
```

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

このセクションでは、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、[https://access.redhat.com/documentation/en-us/openshift\\_container\\_platform/4.3/html-single/registry/architecture-component-imageregistry](https://access.redhat.com/documentation/en-us/openshift_container_platform/4.3/html-single/registry/architecture-component-imageregistry) 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
```

- AWS の場合:

編集前



<pre>. . . storage:   pvc:     claim: registry-cephfs-rwx-pvc . . . .</pre>
編集後
<pre>. . . storage: . . . .</pre>

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。

- VMware の場合:

編集前
<pre>. . . storage:   pvc:     claim: registry-cephfs-rwx-pvc . . . .</pre>
編集後
<pre>. . . storage:   emptyDir: {} . . . .</pre>

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
```

### 3.3. 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 を使用するように設定されている必要があります。

#### 手順

1. namespace にある **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
```