Red Hat OpenShift Data Foundation 4.12

使用 IBM Z 基础架构部署 OpenShift Data Foundation

有关部署 Red Hat OpenShift Data Foundation 以在 IBM Z 基础架构上使用本地存储的说明

Last Updated: 2023-09-19

Red Hat OpenShift Data Foundation 4.12 使用 IBM Z 基础架构部署 OpenShift Data Foundation

有关部署 Red Hat OpenShift Data Foundation 以在 IBM Z 基础架构上使用本地存储的说明

法律通告

摘要

请阅读本文档，介绍如何安装 Red Hat OpenShift Data Foundation 在 IBM Z 基础架构上使用本地存储。 While this document refers only to IBM Z, all information in it also applies to LinuxONE.

目录

使开源包含更多

红帽致力于替换我们的代码、文档和 Web 属性中存在问题的语言。我们从这四个术语开始：master、slave、黑名单和白名单。由于此项工作十分艰巨，这些更改将在即将推出的几个发行版本中逐步实施。详情请查看 CTO Chris Wright 的信息。

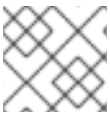
对红帽文档提供反馈

我们感谢您对文档提供反馈信息。请告诉我们如何让它更好。提供反馈：

- 关于特定内容的简单评论：
  1. 请确定您使用 *Multi-page HTML* 格式查看文档。另外，确定 **Feedback** 按钮出现在文档页的右上方。
  2. 用鼠标指针高亮显示您想评论的文本部分。
  3. 点在高亮文本上弹出的 **Add Feedback**。
  4. 按照显示的步骤操作。
- 要提交更复杂的反馈，请创建一个 Bugzilla ticket：
  1. 进入 Bugzilla 网站。
  2. 在 **Component** 部分中，选择 **文档**。
  3. 在 **Description** 中输入您要提供的信息。包括文档相关部分的链接。
  4. 点 **Submit Bug**。

前言

Red Hat OpenShift Data Foundation 支持在连接或断开连接的环境中的现有 Red Hat OpenShift Container Platform (RHOCP) IBM System Z 集群上部署，以及代理环境的开箱即用支持。



注意

如需有关部署要求的更多信息，请参阅规划部署并准备部署 OpenShift Data Foundation。

要部署 OpenShift Data Foundation，请遵循适合您的环境的部署流程：

- 内部附加设备模式
  - 使用本地存储设备部署
- 外部模式

第 1 章 准备部署 OPENSHIFT 数据基础

使用本地存储设备在 OpenShift Container Platform 上部署 OpenShift Data Foundation 时，您可以创建内部集群资源。此方法在内部置备基本服务，所有应用程序都可以访问额外的存储类。

在使用本地存储开始部署 Red Hat OpenShift Data Foundation 前，请确保满足您的资源要求。请参阅使用本地存储设备安装 OpenShift Data Foundation 的要求。

在外部密钥管理系统 (KMS) 上，

- 当为加密选择 Token 验证方法时，请参考使用 KMS 通过 Token 身份验证启用集群范围的加密 。
- 确保您在 Vault 服务器上使用签名的证书。

在解决了以上问题后，按照给出的顺序执行这些步骤：

1. 安装 Red Hat OpenShift Data Foundation Operator 。
2. 安装 Local Storage Operator。
3. 查找可用的存储设备。
4. 在 IBM Z 上创建 OpenShift Data Foundation 集群服务 。

1.1. 使用本地存储设备安装 OPENSHIFT DATA FOUNDATION 的要求

节点要求

集群必须至少包含三个 OpenShift Container Platform worker 节点，每个节点都有本地附加存储设备。

- 在三个选择的节点的每个节点中都至少有一个可用的原始块设备。OpenShift Data Foundation 使用一个或多个可用的原始块设备。
- 您使用的设备必须为空；磁盘中不得包含物理卷 (PV)，卷组 (VG) 或逻辑卷 (LV)。

如需更多信息，请参阅规划指南中的*资源要求*部分。

1.2. 使用 TOKEN 验证方法通过 KMS 启用集群范围的加密

您可以在密码库中启用用于令牌身份验证的键值后端路径和策略。

先决条件

- 管理员对 vault 的访问权限。
- 有效的 Red Hat OpenShift Data Foundation 高级订阅。如需更多信息，请参阅 OpenShift Data Foundation 订阅中的知识库文章。
- 仔细选择唯一路径名称作为遵循命名惯例的后端**路径**，因为它无法在以后更改。

流程

1. 在密码库中启用 Key/Value(KV)后端路径。
   对于 vault KV 机密引擎 API，版本 1：

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

对于 vault KV 机密引擎 API，版本 2：

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

2. 创建策略来限制用户在 secret 上执行写入或删除操作：

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

3. 创建与上述策略匹配的令牌：

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

第 2 章 使用本地存储设备部署 OPENSHIFT DATA FOUNDATION

使用本地存储设备在 OpenShift Container Platform 上部署 OpenShift Data Foundation 为您提供创建内部集群资源的选项。按照这个部署方法，使用本地存储来支持 OpenShift Container Platform 应用程序的持久性卷。

使用本节在已安装 OpenShift Container Platform 的 IBM Z 基础架构上部署 OpenShift Data Foundation。

2.1. 安装 RED HAT OPENSHIFT DATA FOUNDATION OPERATOR

您可以使用 Red Hat OpenShift Container Platform Operator Hub 安装 Red Hat OpenShift Data Foundation Operator。

先决条件

- 使用具有 **cluster-admin** 和 operator 安装权限的账户访问 OpenShift Container Platform 集群。
- 您必须在 Red Hat OpenShift Container Platform 集群中至少有三个 worker 节点。
- 有关其他资源要求，请参阅规划您的部署指南。

重要

- 当您需要覆盖 OpenShift Data Foundation 的集群范围默认节点选择器时，您可以使用以下命令为 **openshift-storage** 命名空间指定空白节点选择器（在这种情况下创建 **openshift-storage** 命名空间）：

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

- 将节点作为 **infra** 污点，以确保只在该节点上调度 Red Hat OpenShift Data Foundation 资源。这有助于您节省订阅成本。如需更多信息，请参阅管理和分配存储资源指南中的*如何将专用 worker 节点用于 Red Hat OpenShift Data Foundation* 部分。

流程

1. 登录 OpenShift Web 控制台。
2. 点 **Operators → OperatorHub**。
3. 在 **Filter by keyword** 框中滚动或键入 **OpenShift Data Foundation**，以查找 **OpenShift Data Foundation** Operator。
4. 点 **Install**。
5. 在 **Install Operator** 页面中设置以下选项：
   a. 更新频道为 **stable-4.12**。
   b. 安装模式是 **A specific namespace on the cluster**。
   c. Installed Namespace 为 **Operator recommended namespace openshift-storage**。如果命名空间 **openshift-storage** 不存在，它会在 Operator 安装过程中创建。

d. 将 Approval Strategy 选为 **Automatic** 或 **Manual**。
如果选择 **Automatic** 更新，Operator Lifecycle Manager(OLM)将自动升级 Operator 的运行实例，而无需任何干预。

如果选择 **手动** 更新，则 OLM 会创建一个更新请求。作为集群管理员，您必须手动批准该更新请求，才能将 Operator 更新至更新的版本。

e. 确保为 **Console 插件** 选择了 **Enable** 选项。

f. 点 **Install**。

验证步骤

- 成功安装 Operator 后，用户界面中会显示一个带有 **Web console update is available** 信息的弹出窗口。点这个弹出窗口中的 **Refresh web console** 来反映控制台的更改。
- 在 Web 控制台中：
  - 进入到 Installed Operators，再验证 **OpenShift Data Foundation** Operator 是否显示绿色勾号，指示安装成功。
  - 进入到 **Storage**，再验证 **Data Foundation** 仪表板是否可用。

2.2. 安装 LOCAL STORAGE OPERATOR

在本地存储设备上创建 Red Hat OpenShift Data Foundation 集群前，请先从 Operator Hub 安装 Local Storage Operator。

流程

1. 登录 OpenShift Web 控制台。
2. 点 **Operators → OperatorHub**。
3. 在 **Filter by keyword** 框中键入 **local storage**，从操作器列表中搜索 **Local Storage operator** 并单击它。
4. 在 **Install Operator** 页面中设置以下选项：
   a. 将频道更新为 **4.12** 或 **stable**。
   b. 安装模式是 **A specific namespace on the cluster**。
   c. Installed Namespace 为 **Operator recommended namespace openshift-local-storage**。
   d. 将批准更新为 **Automatic**。
5. 点 **Install**。

验证步骤

- 验证 Local Storage Operator 是否显示绿色勾号，代表安装成功。

2.3. 查找可用的存储设备（可选）

此步骤是额外的信息，可以在存储集群创建过程中自动发现磁盘时跳过。在为 IBM Z 创建持久性卷(PV)之前，使用 OpenShift Data Foundation 标签 **cluster.ocs.openshift.io/openshift-storage="** 标记的三个或更多 worker 节点的设备名称。

流程

1. 使用 OpenShift Data Foundation 标签列出并验证工作程序节点的名称。

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

输出示例：

```
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. 登录用于 OpenShift Data Foundation 资源的每个 worker 节点，并为每个可用的原始块设备查找唯一的 **by-id** 设备名称。

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

输出示例：

```
$ 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
loop0                          7:0    0   500G  0 loop
sda                            8:0    0   120G  0 disk
|-sda1                         8:1    0   384M  0 part /boot
`-sda4                         8:4    0 119.6G  0 part
`-coreos-luks-root-nocrypt   253:0    0 119.6G  0 dm   /sysroot
sdb                            8:16   0   500G  0 disk
```

在本例中，对于 **bmworker01**，可用的本地设备是 **sdb**。

3. 确定在第 2 步中选择的每个设备的唯一 ID。

```
sh-4.4#ls -l /dev/disk/by-id/  | grep sdb
lrwxrwxrwx. 1 root root  9 Feb  3 16:49 scsi-360050763808104bc2800000000000259 ->
../../sdb
lrwxrwxrwx. 1 root root  9 Feb  3 16:49 scsi-SIBM_2145_00e020412f0aXX00 -> ../../sdb
lrwxrwxrwx. 1 root root  9 Feb  3 16:49 scsi-0x60050763808104bc2800000000000259 ->
../../sdb
```

在上例中，本地设备 **sdb** 的 ID

```
scsi-0x60050763808104bc2800000000000259
```

4. 重复上述步骤，识别所有由 OpenShift Data Foundation 使用存储设备的其他节点的设备 ID。详情请查看本知识库文章。

2.4. 在 IBM Z 上创建 OPENSHIFT DATA FOUNDATION 集群

使用此流程在 IBM Z 上创建 OpenShift Data Foundation 集群。

先决条件

- 确保满足使用本地存储设备安装 OpenShift Data Foundation 的要求部分中的所有要求。
- 您必须最少有三个存储类型和大小相同的 worker 节点（例如 200 GB），才能使用 IBM Z 或 LinuxONE 上的本地存储设备。

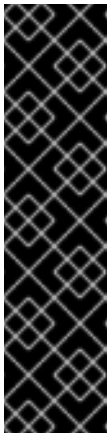
流程

1. 在 OpenShift Web 控制台中，点 **Operators → Installed Operators** 查看所有已安装的 Operator。
   确保所选 **项目** 为 **openshift-storage**。
2. 单击 **OpenShift Data Foundation** 操作器，然后单击 **Create StorageSystem**。
3. 在 Backing storage 页面中，执行以下操作：
   a. 选择 **Create a new StorageClass using the local storage devices for Backing storage type** 选项。
   b. 为**部署类型**选项选择 **Full Deployment**。
   c. 点 Next。

重要

如果还没有安装，系统会提示您安装 Local Storage Operator。点 **Install** 并按照 Installing Local Storage Operator 中所述的步骤进行操作。

4. 在 **Create local volume set** 页面中，提供以下信息：
   a. 为 **LocalVolumeSet** 和 **StorageClass** 输入一个名称。
   默认情况下，存储类名称会出现本地卷集名称。您可以更改名称。
   b. 选择以下任意一项：
   - **所有节点上的磁盘**
     使用与所有节点上所选过滤器匹配的可用磁盘。
   - **所选节点上的磁盘**
     仅在所选节点上使用与所选过滤器匹配的可用磁盘。



重要

- 只有在您使用三个或更多节点创建的存储集群分布到三个可用区的最低要求时，才会启用灵活的扩展功能。
  有关灵活扩展的信息，请参阅知识库文章*在启用灵活扩展时使用 YAML 扩展 OpenShift Data Foundation 集群*。
- 灵活扩展功能会在部署时启用，以后无法启用或禁用。
- 如果选择的节点与 OpenShift Data Foundation 的一个聚合的 30 个 CPU 和 72 GiB RAM 的要求不匹配，则会部署一个最小的集群。
  如需最低起始节点要求，请参阅 *规划指南*中的资源要求部分。

c. 从可用 **Disk Type** 列表中，选择 **SSD/NVME**。

d. 展开 **Advanced** 部分并设置以下选项：

卷模式	默认会选择块。
设备类型	从下拉列表中选择一个或多个设备类型。
磁盘大小	为设备设置最小 100GB 大小，以及需要包含的设备的最大可用大小。
磁盘限制上限	这表示节点上可以创建的 PV 数量上限。如果此字段留空，则为匹配节点上的所有可用磁盘创建 PV。

e. 点 **Next**。
此时会显示一个用于确认创建 LocalVolumeSet 的弹出窗口。

f. 单击 **Yes** 以继续。

5. 在 **Capacity 和 nodes** 页面中，配置以下内容：

a. **可用的原始容量**会根据与存储类关联的所有附加磁盘填充容量值。这将需要一些时间才能出现。**Selected nodes** 列表根据存储类显示节点。

b. 您可以选择 **Taint** 节点复选框。

c. 点 **Next**。

6. 可选：在 **Security and network** 页面中，根据您的要求进行配置：

a. 若要启用加密，可选择为**块存储和文件存储启用数据加密。**

b. 选择以下一个或两个**加密级别** ：

- **集群范围的加密**
  加密整个集群（块和文件）。
- **StorageClass 加密**
  使用启用加密的存储类创建加密的持久性卷（仅块）。

c. 选中**连接到外部密钥管理服务**复选框。这是集群范围加密的可选选项。

i. 默认情况下，**Key Management Service Provider**设置为 **Vault**。

ii. 输入 Vault **Service Name**、Vault 服务器的主机**地址** ('https://*<hostname 或 ip>*')、**端口**号和 **Token**。

iii. 展开 **Advanced Settings** 以根据您的 Vault 配置输入额外的设置和证书详情：

A. 在 **后端路径**中输入为 OpenShift Data Foundation 专用且唯一的 Key Value secret 路径。

B. （可选）输入 **TLS 服务器名称**和 **Vault Enterprise 命名空间**。

C. 上传对应的 PEM 编码证书文件，以提供 **CA 证书**、**客户端证书**和**客户端私钥**。

D. 点 **Save**。

d. 选择 **Default(SDN)**，因为 IBM Z 基础架构上的 OpenShift Data Foundation 尚不支持 Multus。

e. 点 **Next**。

7. 在 Review and create page 中：

a. 检查配置详情。若要修改任何配置设置，请单击 **Back** 以返回到上一配置页面。

b. 单击 **Create StorageSystem**。

验证步骤

- 验证已安装存储集群的最终状态：

a. 在 OpenShift Web 控制台中，导航到 **Installed Operators → OpenShift Data Foundation → Storage System → ocs-storagecluster-storagesystem → Resources**。

b. 验证 **StorageCluster** 的 **Status** 是否为 **Ready**，并且旁边有一个绿色勾号标记。

- 要验证是否在存储集群中启用了灵活的扩展，请执行以下步骤：

1. 在 OpenShift Web 控制台中，导航到 **Installed Operators → OpenShift Data Foundation → Storage System → ocs-storagecluster-storagesystem → Resources → ocs-storagecluster**。

2. 在 YAML 选项卡中，在 **spec** 部分搜索键 **flexibleScaling**，在 **status** 部分搜索 **failureDomain**。如果 **flexible scaling** 为 true，**failureDomain** 被设置为 host，则启用灵活的扩展功能。

```
spec:
flexibleScaling: true
[…]
status:
failureDomain: host
```

- 要验证 OpenShift 数据基础的所有组件是否已成功安装，请参阅验证您的 OpenShift Data Foundation 部署。

其他资源

- 要扩展初始集群的容量，请参阅扩展存储指南。

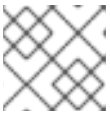
第 3 章 为内部附加设备模式验证 OPENSHIFT DATA FOUNDATION 部署

使用本节验证 OpenShift Data Foundation 是否已正确部署。

3.1. 验证 POD 的状态

流程

1. 从 OpenShift Web 控制台点 **Workloads → Pods**。
2. 从 **Project** 下拉列表中选择 **openshift-storage**。



注意

如果禁用 **Show default projects** 选项，请使用切换按钮列出所有默认项目。

有关每个组件预期的 pod 数量及其变化取决于节点数量的更多信息，请参阅 表 3.1 "对应 OpenShift Data Foundation 集群的 Pod"。

3. 点 **Running** 和 **Completed** 标签页验证以下 pod 是否处于 **Running** 和 **Completed** 状态：

表 3.1. 对应 OpenShift Data Foundation 集群的 Pod

组件	对应的 pod
OpenShift Data Foundation Operator	• **OCS-operator-*** （在任何 worker 节点上有 1 个 pod） • **ocS-metrics-exporter-*** （任何 worker 节点上 1 个 pod） • **odF-operator-controller-manager-*** （任何 worker 节点上 1 个 pod） • **csi-addons-controller-manager-*** （任何 worker 节点上 1 个 pod） • **odf-console-*** (任何 worker 节点上 1 个 pod)
Rook-ceph Operator	**rook-ceph-operator-*** （任何 worker 节点上有 1 个 pod）

组件	对应的 pod
多云对象网关	• **noobaa-operator-*** (任何 worker 节点上 1 个 pod) • **noobaa-core-*** (任何存储节点上 1 个 pod) • **noobaa-db-pg-*** (任何存储节点上 1 个 pod) • **noobaa-endpoint-*** (任何存储节点上 1 个 pod)
MON	**rook-ceph-mon-*** （在存储节点间分布 3 个 pod）
MGR	**rook-ceph-mgr-*** （任何存储节点上的 1 个 pod）
MDS	**rook-ceph-mds-ocs-storagecluster-cephfilesystem-*** （2 个 pod 在存储节点间分布）
RGW	**rook-ceph-rgw-ocs-storagecluster-cephobjectstore-*** （任何存储节点上的 1 个 pod）

CSI	• **cephfs**
	  ◦ **csi-cephfsplugin-*** (每个 worker 节点上 1 个 pod)
	  ◦ **csi-cephfsplugin-provisioner-*** （2 个 pod 在不同的 worker 节点上分布）
	• **rbd**
	  ◦ **csi-rbdplugin-*** (每个 worker 节点上 1 个 pod)
	  ◦ **csi-rbdplugin-provisioner-*** （2 个 pod 在不同的 worker 节点上分步）

rook-ceph-crashcollector	**rook-ceph-crashcollector-*** （每个存储节点上 1 个 pod）

组件	对应的 pod
OSD	• **rook-ceph-osd-*** (每个设备 1 个 pod) • **rook-ceph-osd-prepare-ocs-deviceset-*** （每个设备 1 个 pod）

3.2. 验证 OPENSHIFT DATA FOUNDATION 集群是否健康

流程

1. 在 OpenShift Web 控制台中，点 **Storage → Data Foundation**。
2. 点 **Storage Systems** 选项卡，然后点 **ocs-storagecluster-storagesystem**。
3. 在 Overview 选项卡下的 Block and File 仪表板的 **Status 卡** 中，验证 *Storage Cluster* 和 *Data Resiliency* 都带有绿色勾号标记。
4. 在 **Details 卡** 中，验证是否显示集群信息。

如需有关使用 Block and File 仪表板的 OpenShift Data Foundation 集群健康的更多信息，请参阅监控 OpenShift Data Foundation。

3.3. 验证 MULTICLOUD 对象网关是否健康

流程

1. 在 OpenShift Web 控制台中，点 **Storage → Data Foundation**。
2. 在 **Overview** 选项卡的 **Status** 卡中，点 **Storage System**，然后点弹出框中的存储系统链接。
   a. 在 **Object** 选项卡的 **Status 卡** 中，验证 *Object Service* 和 *数据弹性* 都具有绿色勾号。
   b. 在 **Details** 卡中，验证是否显示了 MCG 信息。

如需有关使用对象服务仪表板的 OpenShift Data Foundation 集群健康的更多信息，请参阅监控 OpenShift Data Foundation。

3.4. 验证特定的存储类是否存在

流程

1. 从 OpenShift Web 控制台左侧窗格中，点击 **Storage → Storage Classes**。
2. 验证是否在创建 OpenShift Data Foundation 集群时创建了以下存储类：
   - **ocs-storagecluster-ceph-rbd**
   - **ocs-storagecluster-cephfs**
   - **openshift-storage.noobaa.io**

- **ocs-storagecluster-ceph-rgw**

第 4 章 卸载 OPENSHIFT DATA FOUNDATION

4.1. 在内部附加设备模式中卸载 OPENSHIFT DATA FOUNDATION

使用本节中的步骤卸载 OpenShift Data Foundation。

卸载注解

Storage Cluster 上的注解用于更改卸载过程的行为。要定义卸载行为，在存储集群中引入了以下两个注解：

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

下表提供了有关可用于这些注解的不同值的信息：

表 4.1. uninstall.ocs.openshift.io 卸载注解描述

注解	值	默认值	行为
cleanup-policy	delete	是	Rook 清理物理驱动器和 **DataDirHostPath**
cleanup-policy	retain	否	Rook **不会**清理物理驱动器和 **DataDirHostPath**
模式	graceful	是	rook 和 NooBaa 在管理员/用户删除持久性卷声明(PVC)和 Object Bucket Claims(OBC)前**暂停**卸载过程。
模式	forced	否	rook 和 NooBaa 会在卸载过程中继续进行，即使 PVC/OBCs 使用 Rook 和 NooBaa 置备

编辑注解的值，以更改清理策略或卸载模式。

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

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

两个命令的预期输出：

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

先决条件

- 确保 OpenShift Data Foundation 集群处于健康状态。当因为资源或节点不足而导致部分 pod 无法成功终止时，卸载过程可能会失败。如果集群处于不健康状态，请在卸载 OpenShift Data Foundation 前联系红帽客户支持。
- 使用 OpenShift Data Foundation 提供的存储类，确保应用程序不使用持久性卷声明 (PVC) 或对象存储桶声明 (OBC)。
- 如果管理员创建了任何自定义资源（如自定义存储类、cephblockpools），则管理员必须在移除消耗这些资源后将它们删除。

流程

1. 删除使用 OpenShift Data Foundation 的卷快照。
   a. 列出来自所有命名空间的卷快照。

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

   b. 从上一命令的输出中，识别和删除使用 OpenShift Data Foundation 的卷快照。

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

   <VOLUME-SNAPSHOT-NAME>
   是卷快照的名称

   <NAMESPACE>
   是项目的命名空间

2. 删除使用 OpenShift Data Foundation 的 PVC 和 OBC。
   在默认的卸载模式 (graceful) 中，卸载程序会等待所有使用 OpenShift Data Foundation 的 PVC 和 OBC 被删除。

   如果要在没有删除 PVC 的情况下删除 Storage Cluster，您可以将卸载模式注解设置为 **forced（强制）** 并跳过此步骤。这样做会在系统中产生孤立 PVC 和 OBC。

   a. 使用 OpenShift Data Foundation 删除 OpenShift Container Platform 监控堆栈 PVC。
   请参阅从 OpenShift Data Foundation 中删除监控堆栈

   b. 使用 OpenShift Data Foundation 删除 OpenShift Container Platform Registry PVC。
   从 OpenShift Data Foundation 中删除 OpenShift Container Platform registry

   c. 使用 OpenShift Data Foundation 删除 OpenShift Container Platform 日志 PVC。
   从 OpenShift Data Foundation 中删除集群日志记录操作器

   d. 删除使用 OpenShift Data Foundation 置备的其他 PVC 和 OBC。

   - 下面是一个示例脚本，用于标识使用 OpenShift Data Foundation 置备的 PVC 和 OBC。该脚本将忽略 OpenShift Data Foundation 内部使用的 PVC。

   ```
   #!/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
```

注意

云平台省略 **RGW_PROVISIONER**。

- 删除 OBC。

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

<obc-name>

是 OBC 的名称

<project-name>

是项目的名称

- 删除 PVC。

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

<pvc-name>

是 PVC 的名称

<project-name>

是项目的名称

注意

确保您已删除了集群中创建的任何自定义后备存储、存储桶类等。

3. 删除 Storage System 对象，并等待相关资源被删除。

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

4. 检查 **uninstall.ocs.openshift.io/cleanup-policy** 是否已设置为 **delete**（默认），并确保其状态为 **Completed**。

```
$ oc get pods -n openshift-storage | grep -i cleanup
```

输出示例：

```
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
```

5. 确认目录 **/var/lib/rook** 现在为空。只有 **uninstall.ocs.openshift.io/cleanup-policy** 注解设置为 **delete**（默认）时，此目录才为空。

```
$ 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
```

6. 如果在安装时启用了加密，请从所有 OpenShift Data Foundation 节点上的 OSD 设备删除 **dm-crypt** 管理的 **device-mapper** 映射。

   a. 创建 **debug** pod 和 **chroot** 到存储节点上的主机。

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

```
$ chroot /host
```

   <node-name>
   是节点的名称

   b. 获取设备名称并记录 OpenShift Data Foundation 设备。

```
$ dmsetup ls
```

   输出示例：

```
ocs-deviceset-0-data-0-57snx-block-dmcrypt (253:1)
```

   c. 删除映射的设备。

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

重要

如果上述命令因为权限不足而卡住，请运行以下命令：

- 按 **CTRL+Z** 退出上述命令。
- 查找阻塞的进程的 PID。

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

- 使用 **kill** 命令终止进程。

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

<PID>
是进程 ID

- 验证设备名称是否已移除。

```
$ dmsetup ls
```

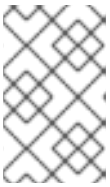
7. 删除命名空间并等待删除完成。如果 **openshift-storage** 是活跃的项目，则需要切换到另一个项目。
例如：

```
$ oc project default
```

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

如果以下命令返回 NotFound 错误，则项目被删除。

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



注意

卸载 OpenShift Data Foundation 时，如果还没有完全删除**命名空间**并处于 **Terminating** 状态，请执行故障排除和删除 Uninstall 期间剩余的资源 的步骤，以识别阻止命名空间终止的对象。

8. 如果您使用本地存储设备部署了 OpenShift Data Foundation，请删除本地存储 Operator 配置。请参阅 删除本地存储 Operator 配置。

9. 取消标记存储节点。

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

```
$ oc label nodes  --all topology.rook.io/rack-
```

10. 如果节点存在污点，则删除 OpenShift Data Foundation 污点。

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

11. 确认已删除使用 OpenShift Data Foundation 置备的所有持久卷(PV)。如果有任何 PV 处于 **Released** 状态，请将其删除。

```
$ oc get pv
```

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

<pv-name>
是 PV 的名称

12. 删除 **CustomResourceDefinitions**。

```
$ 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 storagesystems.odf.openshift.io --
wait=true --timeout=5m
```

13. 要确保在 OpenShift Container Platform Web 控制台中完全卸载 OpenShift Data Foundation，

    a. 点 **Storage**。

    b. 验证 **OpenShift Data Foundation** 是否不再出现在 Storage 下。

4.1.1. 删除本地存储 Operator 配置

只有在您使用本地存储设备部署了 OpenShift Data Foundation 时，才使用本节中的说明。

注意

对于仅使用 **localvolume** 资源部署的 OpenShift Data Foundation，请直接转到第 8 步。

流程

1. 标识 **LocalVolumeSet** 以及 OpenShift Data Foundation 使用的对应 **StorageClassName**。

```
$ oc get localvolumesets.local.storage.openshift.io -n openshift-local-storage
```

2. 将变量 SC 设置为提供 **LocalVolumeSet** 的 **StorageClass**。

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

3. 列出并记下稍后要清理的设备。要列出磁盘的设备 ID，请按照这里所述的步骤进行操作，请参阅查找可用的存储设备。
输出示例：

```
/dev/disk/by-id/scsi-360050763808104bc28000000000000eb
/dev/disk/by-id/scsi-360050763808104bc28000000000000ef
/dev/disk/by-id/scsi-360050763808104bc28000000000000f3
```

4. 删除 **LocalVolumeSet**。

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

5. 删除给定 **StorageClassName** 的本地存储 PV。

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

6. 删除 **StorageClassName**。

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

7. 删除 **LocalVolumeSet** 创建的符号链接。

```
[[ ! -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
```

8. 删除 **LocalVolumeDiscovery**。

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

9. 删除 **LocalVolume** 资源（如果有）。
使用以下步骤删除在当前或以前的 OpenShift Data Foundation 版本中置备 PV 的 **LocalVolume** 资源。此外，确保这些资源不提供给集群上的其他租户使用。

   对于每个本地卷，请执行以下操作：

   a. 标识 **LocalVolume** 以及 OpenShift Data Foundation 使用的对应 **StorageClassName**。

   ```
   $ oc get localvolume.local.storage.openshift.io -n openshift-local-storage
   ```

   b. 将变量 LV 设置为 LocalVolume 的名称，变量 SC 设置为 StorageClass 的名称
   例如：

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

   c. 列出并记下稍后要清理的设备。

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

   输出示例：

   ```
   /dev/sdb
   /dev/sdc
   /dev/sdd
   /dev/sde
   ```

d. 删除本地卷资源。

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

e. 删除剩余的 PV 和 StorageClasses（如果存在）。

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

f. 从该资源的存储节点中清理工件。

```
$ [[ ! -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
```

输出示例：

```
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 ...
```

10. 分别擦除第 1 和第 8 步中列出的每个本地卷组或本地卷的磁盘，以便可以重复使用它们。

a. 列出存储节点。

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

输出示例：

```
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. 获取节点控制台并在出现提示时执行 **chroot /host** 命令。

```
$ 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. 将磁盘路径存储在 **DISKS** 变量中的引号内。有关磁盘路径列表，请查看第 3 步和第 8.c 步以了解本地卷集和本地卷。
输出示例：

```
sh-4.4# DISKS="/dev/disk/by-id/scsi-360050763808104bc28000000000000eb
/dev/disk/by-id/scsi-360050763808104bc28000000000000ef /dev/disk/by-id/scsi-
360050763808104bc28000000000000f3 "
or
sh-4.2# DISKS="/dev/sdb /dev/sdc /dev/sdd /dev/sde ".
```

d. 在所有磁盘上运行 **sgdisk --zap-all**。

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

输出示例：

```
Creating new GPT entries.
GPT data structures destroyed! You may now partition the disk using fdisk or
other utilities.
Creating new GPT entries.
GPT data structures destroyed! You may now partition the disk using fdisk or
other utilities.
Creating new GPT entries.
GPT data structures destroyed! You may now partition the disk using fdisk or
other utilities.
Creating new GPT entries.
GPT data structures destroyed! You may now partition the disk using fdisk or
other utilities.
```

e. 退出 shell，再对其他节点重复此操作。

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

11. 删除 **openshift-local-storage** 命名空间并等待删除完成。如果 **openshift-local-storage** 命名空间是活跃的项目，则需要切换到另一个项目。
例如：

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

如果以下命令返回了 **NotFound** 错误，则该项目将被删除。

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

4.2. 从 OPENSHIFT DATA FOUNDATION 中删除监控堆栈

使用本节清理 OpenShift Data Foundation 的监控堆栈。

在配置监控堆栈时创建的 PVC 位于 **openshift-monitoring** 命名空间中。

先决条件

- PVC 被配置为使用 OpenShift Container Platform 监控堆栈。
  如需更多信息，请参阅配置监控堆栈。

流程

1. 列出当前在 **openshift-monitoring** 命名空间中运行的 pod 和 PVC。

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

输出示例：

```
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. 编辑监控 **configmap**。

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

删除引用 OpenShift Data Foundation 存储类的所有 **config** 部分，如下例所示并保存。

编辑前

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

编辑后

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

在本例中，**alertmanagerMain** 和 **prometheusK8s** 监控组件使用 OpenShift Data Foundation PVC。

3. 删除相关的 PVC。请确定删除所有消耗存储类的 PVC。

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

<pvc-name>

是 PVC 的名称

4.3. 从 OPENSHIFT DATA FOUNDATION 中删除 OPENSHIFT CONTAINER PLATFORM REGISTRY

使用这个部分从 OpenShift Data Foundation 清理 OpenShift Container Platform registry。如果要配置其他存储，请参阅 镜像 registry。

在配置 OpenShift Container Platform registry 时创建的 PVC 位于 **openshift-image-registry** 命名空间中。

先决条件

- 镜像 registry 必须已配置为使用 OpenShift Data Foundation PVC。

流程

1. 编辑 **configs.imageregistry.operator.openshift.io** 对象，并删除 **storage** 部分中的内容。

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

编辑前

```
.
.
.

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

编辑后

```
.
.
.
storage:
  emptyDir: {}
.
.
.
```

在本例中，PVC 称为 **registry-cephfs-rwx-pvc**，现在可以安全地删除。

2. 删除 PVC。

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

<pvc-name>
是 PVC 的名称

4.4. 从 OPENSHIFT DATA FOUNDATION 中删除集群日志记录操作器

使用本节从 OpenShift Data Foundation 清理集群日志记录 Operator。

在配置集群日志记录 Operator 时创建的 PVC 位于 **openshift-logging** 命名空间中。

先决条件

- 集群日志记录实例应该已配置为使用 OpenShift Data Foundation PVC。

流程

1. 删除命名空间中的 **ClusterLogging** 实例。

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

openshift-logging 命名空间中的 PVC 现在可以安全地删除。

2. 删除 PVC。

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

<pvc-name>

是 PVC 的名称