Red Hat OpenShift Data Foundation 4.12

为 OpenShift Workloads 配置 OpenShift Data Foundation 灾难恢复

Red Hat Advanced Cluster Management for Kubernetes 2.7 的 Red Hat OpenShift Data Foundation Metro-DR 功能现已正式发布，且 Blocks 和 Files 的 Regional-DR 解决方案作为技术预览提供，并按技术预览进行支持。

Last Updated: 2023-12-21

Red Hat OpenShift Data Foundation 4.12 为 OpenShift Workloads 配置 OpenShift Data Foundation 灾难恢复

Red Hat Advanced Cluster Management for Kubernetes 2.7 的 Red Hat OpenShift Data Foundation Metro-DR 功能现已正式发布，且 Blocks 和 Files 的 Regional-DR 解决方案作为技术预览提供，并按技术预览进行支持。

法律通告

摘要

本指南旨在详细说明使用 Advanced Cluster Management 部署 OpenShift Data Foundation 进行灾难恢复所需的步骤，以实现高度可用的存储基础架构。

目录

让开源更具包容性

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。

要提供反馈，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**。

第 1 章 OPENSHIFT DATA FOUNDATION 灾难恢复简介

灾难恢复（DR）是从自然或人为的灾难中恢复并继续业务关键应用程序的能力。它是任何主要组织的整体业务整合战略的一个组件，其设计旨在在重要事件期间保持业务操作的连续性。

OpenShift Data Foundation DR 功能可在多个 Red Hat OpenShift Container Platform 集群间启用 DR，并分类如下：

- **Metro-DR**
  Metro-DR 可确保在数据中心出现问题时保持业务的连续性，并不会造成数据丢失。在公有云中，它们类似于防止可用性区域失败。

- **Regional-DR**
  区域 DR 可以在一个地理区域出现问题时确保业务的连续性（在可以接受一些可预测数量的数据丢失的情况下）。在公有云中，它们类似于防止区域故障。

Metro-DR 中的区故障以及 Regional-DR 中的区域故障通常使用术语 **Recovery Point Objective (RPO)** 和 **Recovery Time Objective (RTO)**。

- **RPO** 是一种衡量持久性数据备份或快照的频率。实际上，RPO 表示在中断后将丢失或需要重新输入的数据量。

- **RTO** 是企业可以容忍的停机时间。RTO 回答了这个问题，"在收到业务中断通知后，我们的系统需要多久才能恢复？"

本指南旨在详细介绍灾难恢复步骤和命令，将应用程序从一个 OpenShift Container Platform (OCP) 集群切换到另一个集群，然后将同一应用程序恢复到原始集群。

第 2 章 灾难恢复订阅要求

Red Hat OpenShift Data Foundation 支持的灾难恢复功能需要满足以下所有先决条件，才能成功实施灾难恢复解决方案：

- 有效的 Red Hat OpenShift Data Foundation 高级授权
- 有效的 Red Hat Advanced Cluster Management for Kubernetes 订阅

任何包含 PV（包括作为源或目标）的 PV 的 Red Hat OpenShift Data Foundation 集群都需要 OpenShift Data Foundation 高级授权。此订阅应该在源和目标集群上处于活跃状态。

要了解 OpenShift Data Foundation 订阅如何工作，请参阅与 OpenShift Data Foundation 订阅相关的知识库文章。

第 3 章 用于 OPENSHIFT DATA FOUNDATION 的 METRO-DR 解决方案

本指南详细介绍了区域灾难恢复 (Metro DR) 步骤和命令，可以将应用程序从一个 OpenShift Container Platform 集群切换到另一个集群，然后将同一应用程序恢复到原始集群。在这种情况下，会使用 Red Hat Advanced Cluster Management (RHACM) 创建或导入 OCP 集群，且各个 OCP 集群间的距离小于 10ms RTT 延迟。

应用程序的持久存储将由外部 Red Hat Ceph Storage (RHCS) 集群提供，在连接此存储集群的 OpenShift Container Platform 实例的两个位置之间扩展。在站点出现问题时，在第三个位置中（与部署 OpenShift Container Platform 实例不同的位置）需要一个带有存储监控器服务的仲裁节点用于为 RHCS 集群创建仲裁。第三个位置可以位于连接到 OpenShift Container Platform 实例的存储集群中的 ~100ms RTT 范围。

这是在由距离分开的两个不同 OpenShift Container Platform 集群中使用 OpenShift Data Foundation 和 RHACM 配置和执行 OpenShift 灾难恢复 (ODR) 功能所需的 Metro DR 步骤的一般概述。除了这两个被称为受管集群的集群外，还需要第三个 OpenShift Container Platform 集群，它将是 Red Hat Advanced Cluster Management (RHACM) hub 集群。

3.1. METRO-DR 解决方案的组件

Metro-DR 由 Red Hat Advanced Cluster Management for Kubernetes、Red Hat Ceph Storage 和 OpenShift Data Foundation 组件组成，以便在 OpenShift Container Platform 集群中提供应用程序和数据移动性。

Red Hat Advanced Cluster Management for Kubernetes

Red Hat Advanced Cluster Management (RHACM)提供了管理多个集群和应用程序生命周期的功能。因此，它充当多集群环境中的控制平面。

RHACM 分为两个部分：

- RHACM Hub：在多集群 control plane 上运行的组件
- 受管集群：在受管理的集群中运行的组件

有关此产品的更多信息，请参阅 RHACM 文档 和 RHACM"管理应用程序"文档。

Red Hat Ceph Storage

Red Hat Ceph Storage 是一个可大规模扩展、开放、软件定义的存储平台，它将最稳定版本的 Ceph 存储系统与 Ceph 管理平台、部署实用程序和支持服务相结合。它显著降低了存储企业数据的成本，帮助组织管理指数级数据增长。此软件是一款强大、现代化的 PB 级存储平台，适用于公有云或私有云部署。

如需更多信息，请参阅 Red Hat Ceph Storage 。

OpenShift Data Foundation

OpenShift Data Foundation 为 OpenShift Container Platform 集群中有状态应用程序提供部署和管理存储的功能。它由 Ceph 作为存储提供商提供支持，其生命周期由 OpenShift Data Foundation 组件堆栈和 Ceph-CSI 管理，它们的生命周期为有状态应用提供持久卷的调配和管理。

OpenShift DR

OpenShift DR 是跨一组使用 RHACM 部署和管理的有状态应用程序的灾难恢复编排器，并提供云原生接口来编排应用程序状态在持久性卷上的生命周期。它们是：

- 保护跨 OpenShift 集群的应用程序及其状态关系

- 将一个应用程序及其状态转移到一个对等集群
- 将一个应用程序及其状态重新定位到之前部署的集群

OpenShift DR 被分成三个组件：

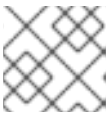
- **ODF Multicluster Orchestrator**: 安装在多集群控制平面 (RHACM Hub) 中，它会编配配置并针对 Metro 和 Regional D 关系对等 OpenShift Data Foundation 集群。
- **OpenShift DR Hub Operator**：在 hub 集群上作为 ODF 多集群安装的一部分自动安装，以编配启用 DR 应用程序的故障转移或重新定位。
- **OpenShift DR Cluster Operator**：在属于 Metro 和 Regional DR relationship的每个受管集群中自动安装，用于管理应用程序的所有 PVC 的生命周期。

3.2. METRO-DR 部署工作流

本节概述使用最新版本的 Red Hat OpenShift Data Foundation、Red Hat Ceph Storage (RHCS)和 Red Hat Advanced Cluster Management for Kubernetes (RHACM) 版本 2.7 或更高版本在两个不同的 OpenShift Container Platform 集群中配置和部署自治功能所需的步骤。除了两个受管集群外，还需要第三个 OpenShift Container Platform 集群来部署 Advanced Cluster Management。

要配置基础架构，请按照给定的顺序执行以下步骤：

1. 确保满足作为 DR 解决方案的三个集群（Hub、Primary 和 Secondary Openshift Container Platform）集群的要求。请参阅启用 Metro-DR 的要求。
2. 确保您具有仲裁者部署 Red Hat Ceph Storage 扩展集群的要求。请参阅部署 Red Hat Ceph Storage 的要求。
3. 部署和配置 Red Hat Ceph Storage 扩展模式。有关使用扩展模式功能在两个不同的数据中心中启用 Ceph 集群的说明，请参阅部署 Red Hat Ceph Storage 。
4. 安装 OpenShift Data Foundation operator，并在 Primary 和 Secondary 受管集群上创建存储系统。请参阅在受管集群中安装 OpenShift Data Foundation。
5. 在 Hub 集群上安装 ODF Multicluster Orchestrator。请参阅在 Hub 集群上安装 ODF 多集群编排器。
6. 配置 Hub、Primary 和 Secondary 集群之间的 SSL 访问。请参阅配置跨集群的 SSL 访问。
7. 创建 DRPolicy 资源，以用于在跨 Primary 和 Secondary 应用程序所需的 DR 保护的应用程序。请参阅在 Hub 集群上创建灾难恢复策略。



注意

Metro-DR 解决方案只能有一个 DRpolicy。

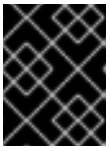
测试您的灾难恢复解决方案：

- 使用 RHACM 控制台创建示例应用程序。请参阅创建示例应用程序。
- 在受管集群之间使用示例应用程序测试故障切换和重定位操作。参阅应用程序故障切换和重新定位应用程序。

3.3. 启用 METRO-DR 的要求

Red Hat OpenShift Data Foundation 支持的灾难恢复功能需要满足以下所有先决条件，才能成功实施灾难恢复解决方案：

- 您必须有以下在它们之间具有网络可访问性的 OpenShift 集群：
  - **Hub 集群**，Red Hat Advanced Cluster Management for Kubernetes (RHACM operator) 在其中安装。
  - **Primary 受管集群**，OpenShift Data Foundation 在其中安装。
  - **Secondary 受管集群**，OpenShift Data Foundation 在其中安装。
- 确保在 Hub 集群中安装 RHACM 操作符和 MultiClusterHub。具体步骤请查看 RHACM 安装指南。
  成功安装 Operator 后，用户界面中会显示一个带有 Web 控制台更新可用消息的弹出窗口。点此弹出窗口中的 **Refresh Web 控制台**来反映控制台更改。



重要

用户负责确保正确配置应用流量路由和重定向。目前不支持对应用程序流量路由的配置和更新。

- 在 Hub 集群中，进入到 **All Clusters → Infrastructure → Clusters**。使用 RHACM 控制台导入或创建**主（Primary）受管集群**和**二级（Secondary）受管集群**。为您的环境选择适当的选项。成功创建或导入受管集群后，您可以看到在控制台中导入或创建的集群列表。具体步骤请参阅创建集群和将目标受管集群导入到 hub 集群。

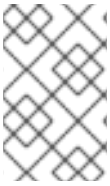
警告

OpenShift Container Platform 受管集群和 Red Hat Ceph Storage (RHCS)节点的位置之间存在距离的限制。站点之间的网络延迟必须低于 10 毫秒往返时间 (RTT)。

3.4. 使用仲裁器部署 RED HAT CEPH STORAGE 的要求

Red Hat Ceph Storage 是一个开源企业平台，可为标准、经济的服务器和磁盘提供统一软件定义型存储。通过将块、对象和文件存储整合为一个平台，Red Hat Ceph Storage 可以高效并自动管理所有数据，因此您可以专注于使用它的应用程序和工作负载。

本节提供了 Red Hat Ceph Storage 部署的基本概述。如需更复杂的部署，请参阅 Red Hat Ceph Storage 5 的官方文档指南。



注意

仅支持 Flash 介质，因为它在降级时使用 **min_size=1**。仅对 all-flash OSD 使用扩展模式。使用 all-flash OSD 会最大程度减少在恢复连接后恢复所需的时间，从而尽量减少数据丢失的可能性。

重要

纠删代码池无法用于扩展模式。

3.4.1. 硬件要求

如需有关部署 Red Hat Ceph Storage 的最低硬件要求的信息，请参阅容器化 Ceph 的最低硬件建议。

表 3.1. Red Hat Ceph Storage 集群部署的物理服务器位置和 Ceph 组件布局：

节点名	数据中心	Ceph 组件
ceph1	DC1	OSD+MON+MGR
ceph2	DC1	OSD+MON
ceph3	DC1	OSD+MDS+RGW
ceph4	DC2	OSD+MON+MGR
ceph5	DC2	OSD+MON
ceph6	DC2	OSD+MDS+RGW
ceph7	DC3	MON

3.4.2. 软件要求

使用 **Red Hat Ceph Storage 5**的最新软件版本。

有关 Red Hat Ceph Storage 支持的操作系统版本的更多信息，请参阅 Red Hat Ceph Storage: 支持的配置。

3.4.3. 网络配置要求

推荐的 Red Hat Ceph Storage 配置如下：

- 您必须有两个独立的网络，一个公共网络和一个专用网络。
- 您必须有三个不同的数据中心，支持所有数据中心的 Ceph 私有和公共网络的 VLAN 和子网。

注意

您可以为每个数据中心使用不同的子网。

- 运行 Red Hat Ceph Storage Object Storage Devices (OSD) 的两个数据中心之间的延迟不能超过 10 ms RTT。对于 **arbiter** 数据中心，这已进行了测试，其值为 100 ms RTT 到其他两个 OSD 数据中心。

以下是我们在本指南中使用的基本网络配置示例：

- **DC1: Ceph public/private network:**10.0.40.0/24
- **DC2: Ceph public/private network:**10.0.40.0/24
- **DC3: Ceph public/private network:**10.0.40.0/24

有关所需网络环境的更多信息，请参阅 Ceph 网络配置。

3.5. 部署 RED HAT CEPH STORAGE

3.5.1. 节点预部署步骤

在安装 Red Hat Ceph Storage Ceph 集群前，请执行以下步骤来满足所有必要的要求。

1. 将所有节点注册到 Red Hat Network 或 Red Hat Satellite 中，并订阅到有效的池：

```
subscription-manager register
subscription-manager subscribe --pool=8a8XXXXXX9e0
```

2. 为以下软件仓库启用 Ceph 集群中的所有节点的访问权限：

   - **rhel-8-for-x86_64-baseos-rpms**
   - **rhel-8-for-x86_64-appstream-rpms**

```
subscription-manager repos --disable="*" --enable="rhel-8-for-x86_64-baseos-rpms" --
enable="rhel-8-for-x86_64-appstream-rpms"
```

3. 如果需要，将操作系统 RPM 更新至最新版本并重新引导：

```
dnf update -y
reboot
```

4. 从集群中选择节点作为 bootstrap 节点。**ceph1** 是本例中的 bootstrap 节点。
   仅在 bootstrap 节点 **ceph1** 上，启用 **ansible-2.9-for-rhel-8-x86_64-rpms** 和 **rhceph-5-tools-for-rhel-8-x86_64-rpms** 存储库：

```
subscription-manager repos --enable="ansible-2.9-for-rhel-8-x86_64-rpms" --
enable="rhceph-5-tools-for-rhel-8-x86_64-rpms"
```

5. 在所有主机中使用裸机/短主机名配置**主机名**。

```
hostnamectl set-hostname <short_name>
```

6. 使用 cephadm 验证用于部署 Red Hat Ceph Storage 的主机名配置。

```
$ hostname
```

   输出示例：

```
ceph1
```

7. 使用 DNS 域名设置 DOMAIN 变量，修改 /etc/hosts 文件并将 fqdn 条目添加到 127.0.0.1 IP。

```
DOMAIN="example.domain.com"

cat <<EOF >/etc/hosts
127.0.0.1 $(hostname).${DOMAIN} $(hostname) localhost localhost.localdomain localhost4
localhost4.localdomain4
::1       $(hostname).${DOMAIN} $(hostname) localhost6 localhost6.localdomain6
EOF
```

8. 使用 **hostname -f** 选项通过 **fqdn** 检查长主机名。

```
$ hostname -f
```

输出示例：

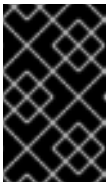
```
ceph1.example.domain.com
```

注：要了解更多有关需要这些更改的信息，请参阅完全限定域名和裸机主机名。

9. 在 bootstrap 节点上执行以下步骤。在我们的示例中，bootstrap 节点为 **ceph1**。

   a. 安装 **cephadm-ansible** RPM 软件包：

```
$ sudo dnf install -y cephadm-ansible
```



重要

要运行 ansible playbook，您必须有 **ssh** 免密码访问配置 Red Hat Ceph Storage 集群的所有节点。确保配置的用户（如 **deployment-user**）具有可调用 **sudo** 命令的 root 特权，而无需输入密码。

   b. 要使用自定义密钥，请配置所选用户（如 **deployment-user**）ssh 配置文件以指定将用于通过 ssh 连接到节点的 id/key：

```
cat <<EOF > ~/.ssh/config
Host ceph*
   User deployment-user
   IdentityFile ~/.ssh/ceph.pem
EOF
```

   c. 构建 ansible 清单

```
cat <<EOF > /usr/share/cephadm-ansible/inventory
ceph1
ceph2
ceph3
ceph4
ceph5
ceph6
ceph7
[admin]
```

```
ceph1
ceph4
EOF
```

注意

此处，属于两个不同的数据中心的主机 (**Ceph1** and **Ceph4**) 配置为清单文件中的 [admin] 组的一部分，并使用 **cephadm** 标记为 **_admin**。每个管理节点都会在 bootstrap 过程中收到 admin ceph 密钥环，以便在一个数据中心停机时，我们都可以使用其他可用的管理节点进行检查。

d. 在运行 pre-flight playbook 前，验证 **ansible** 是否可以使用 ping 模块访问所有节点。

```
$ ansible -i /usr/share/cephadm-ansible/inventory -m ping all -b
```

输出示例：

```
ceph6 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/libexec/platform-python"
    },
    "changed": false,
    "ping": "pong"
}
ceph4 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/libexec/platform-python"
    },
    "changed": false,
    "ping": "pong"
}
ceph3 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/libexec/platform-python"
    },
    "changed": false,
    "ping": "pong"
}
ceph2 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/libexec/platform-python"
    },
    "changed": false,
    "ping": "pong"
}
ceph5 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/libexec/platform-python"
    },
    "changed": false,
    "ping": "pong"
}
ceph1 | SUCCESS => {
    "ansible_facts": {
```

```
        "discovered_interpreter_python": "/usr/libexec/platform-python"
    },
    "changed": false,
    "ping": "pong"
}
ceph7 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/libexec/platform-python"
    },
    "changed": false,
    "ping": "pong"
}
```

e. 进入 **/usr/share/cephadm-ansible** 目录。

f. 使用相对文件路径运行 ansible-playbook。

```
$ ansible-playbook -i /usr/share/cephadm-ansible/inventory /usr/share/cephadm-ansible/cephadm-preflight.yml --extra-vars "ceph_origin=rhcs"
```

preflight playbook Ansible playbook 配置 RHCS **dnf** 存储库，并为引导准备存储集群。它还安装 podman、lvm2、chronyd 和 cephadm。**cephadm-ansible** 和 **cephadm-preflight.yml** 的默认位置为 **/usr/share/cephadm-ansible**。如需更多信息，请参阅运行 preflight playbook

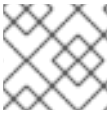
3.5.2. 使用 Cephadm 进行集群引导和服务部署

cephadm 实用程序将安装并启动一个单独的 Ceph Monitor 守护进程，以及运行 cephadm bootstrap 命令本地节点上的新 Red Hat Ceph Storage 集群的 Ceph 管理器守护进程。

在本指南中，我们将使用集群规格 yaml 文件，在一个步骤中使用集群规格 yaml 来部署所有需要的 Red Hat Ceph Storage 服务。

如果您在部署过程中发现问题，通过将部署分成两个步骤来更加轻松地对错误进行故障排除：

1. bootstrap
2. 服务部署



注意

有关 bootstrap 过程的更多信息，请参阅引导新存储集群。

步骤

1. 使用 json 文件创建 json 文件以针对容器 registry 进行身份验证，如下所示：

```
$ cat <<EOF > /root/registry.json
{
 "url":"registry.redhat.io",
 "username":"User",
 "password":"Pass"
}
EOF
```

2. 创建一个 **cluster-spec.yaml**，将节点添加到 Red Hat Ceph Storage 集群，并为服务设置应当运行下表 3.1 的特定标签。

```
cat <<EOF > /root/cluster-spec.yaml
service_type: host
addr: 10.0.40.78  ## <XXX.XXX.XXX.XXX>
hostname: ceph1   ##  <ceph-hostname-1>
location:
  root: default
  datacenter: DC1
labels:
  - osd
  - mon
  - mgr
---
service_type: host
addr: 10.0.40.35
hostname: ceph2
location:
  datacenter: DC1
labels:
  - osd
  - mon
---
service_type: host
addr: 10.0.40.24
hostname: ceph3
location:
  datacenter: DC1
labels:
  - osd
  - mds
  - rgw
---
service_type: host
addr: 10.0.40.185
hostname: ceph4
location:
  root: default
  datacenter: DC2
labels:
  - osd
  - mon
  - mgr
---
service_type: host
addr: 10.0.40.88
hostname: ceph5
location:
  datacenter: DC2
labels:
  - osd
  - mon
---
service_type: host
addr: 10.0.40.66
```

```
hostname: ceph6
location:
  datacenter: DC2
labels:
  - osd
  - mds
  - rgw
---
service_type: host
addr: 10.0.40.221
hostname: ceph7
labels:
  - mon
---
service_type: mon
placement:
  label: "mon"
---
service_type: mds
service_id: cephfs
placement:
  label: "mds"
---
service_type: mgr
service_name: mgr
placement:
  label: "mgr"
---
service_type: osd
service_id: all-available-devices
service_name: osd.all-available-devices
placement:
  label: "osd"
spec:
  data_devices:
    all: true
---
service_type: rgw
service_id: objectgw
service_name: rgw.objectgw
placement:
  count: 2
  label: "rgw"
spec:
  rgw_frontend_port: 8080
EOF
```

3. 使用从 bootstrap 节点配置的 Red Hat Ceph Storage 公共网络，检索 NIC 的 IP。将 **10.0.40.0** 替换为您在 ceph 公共网络中定义的子网后，执行以下命令。

```
$ ip a | grep 10.0.40
```

输出示例：

```
10.0.40.78
```

4. 以 **root** 用户身份在将作为集群中初始 monitor 节点的节点运行 **Cephadm** bootstrap 命令。**IP_ADDRESS** 选项是您用于运行 **cephadm bootstrap** 命令的节点 IP 地址。

注意

如果您配置了不同的用户而不是 **root** 进行免密码 SSH 访问，则使用带有 **cepadm bootstrap** 命令的 **--ssh-user=** 标志。

如果您使用非 default/id_rsa ssh 密钥名称，请使用 **--ssh-private-key** 和 **--ssh-public-key** 选项及 **cephadm** 命令。

```
$ cephadm  bootstrap --ssh-user=deployment-user --mon-ip 10.0.40.78 --apply-spec
/root/cluster-spec.yaml --registry-json /root/registry.json
```

重要

如果本地节点使用完全限定域名 (FQDN)，则将 **--allow-fqdn-hostname** 选项添加到命令行上的 **cephadm bootstrap**。

bootstrap 完成后，您将看到来自之前 cephadm bootstrap 命令的以下输出：

```
You can access the Ceph CLI with:

 sudo /usr/sbin/cephadm shell --fsid dd77f050-9afe-11ec-a56c-029f8148ea14 -c
/etc/ceph/ceph.conf -k /etc/ceph/ceph.client.admin.keyring

Please consider enabling telemetry to help improve Ceph:

 ceph telemetry on

For more information see:

 https://docs.ceph.com/docs/pacific/mgr/telemetry/
```

5. 使用 ceph1 中的 ceph cli 客户端，验证 Red Hat Ceph Storage 集群部署的状态：

```
$ ceph -s
```

输出示例：

```
cluster:
  id:     3a801754-e01f-11ec-b7ab-005056838602
  health: HEALTH_OK

services:
  mon: 5 daemons, quorum ceph1,ceph2,ceph4,ceph5,ceph7 (age 4m)
  mgr: ceph1.khuuot(active, since 5m), standbys: ceph4.zotfsp
  osd: 12 osds: 12 up (since 3m), 12 in (since 4m)
  rgw: 2 daemons active (2 hosts, 1 zones)

data:
  pools:   5 pools, 107 pgs
```

```
objects: 191 objects, 5.3 KiB
usage:   105 MiB used, 600 GiB / 600 GiB avail
         105 active+clean
```

注意

启动所有服务可能需要几分钟时间。

在您未配置任何 osds 时出现一个全局恢复事件是正常的。

您可以使用 **ceph orch ps** 和 **ceph orch ls** 来进一步检查服务的状态。

6. 验证所有节点是否是 **cephadm** 集群的一部分。

```
$ ceph orch host ls
```

输出示例：

```
HOST   ADDR         LABELS  STATUS
ceph1  10.0.40.78    _admin osd mon mgr
ceph2  10.0.40.35    osd mon
ceph3  10.0.40.24    osd mds rgw
ceph4  10.0.40.185   osd mon mgr
ceph5  10.0.40.88    osd mon
ceph6  10.0.40.66    osd mds rgw
ceph7  10.0.40.221   mon
```

注意

您可以直接从主机运行 Ceph 命令，因为 **ceph1** 在 **cephadm-ansible** 清单中配置，作为 [admin] 组的一部分。Ceph 管理密钥在 **cephadm bootstrap** 过程中复制到主机。

7. 检查数据中心的 Ceph 监控服务的当前位置。

```
$ ceph orch ps | grep mon | awk '{print $1 " " $2}'
```

输出示例：

```
mon.ceph1 ceph1
mon.ceph2 ceph2
mon.ceph4 ceph4
mon.ceph5 ceph5
mon.ceph7 ceph7
```

8. 检查数据中心的 Ceph 管理器服务的当前位置。

```
$ ceph orch ps | grep mgr | awk '{print $1 " " $2}'
```

输出示例：

```
mgr.ceph2.ycgwyz ceph2
mgr.ceph5.kremtt ceph5
```

9. 检查 ceph osd crush map 布局，以确保每个主机都配置了 OSD，其状态为 **UP**。此外，再次检查每个节点在表 3.1 中指定的右侧数据中心 bucket 下。

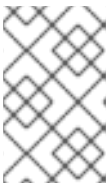
```
$ ceph osd tree
```

输出示例：

```
ID   CLASS  WEIGHT   TYPE NAME         STATUS  REWEIGHT  PRI-AFF
-1         0.87900  root default
-16        0.43950      datacenter DC1
-11        0.14650          host ceph1
 2    ssd  0.14650              osd.2       up   1.00000  1.00000
 -3        0.14650          host ceph2
 3    ssd  0.14650              osd.3       up   1.00000  1.00000
-13        0.14650          host ceph3
 4    ssd  0.14650              osd.4       up   1.00000  1.00000
-17        0.43950      datacenter DC2
 -5        0.14650          host ceph4
 0    ssd  0.14650              osd.0       up   1.00000  1.00000
 -9        0.14650          host ceph5
 1    ssd  0.14650              osd.1       up   1.00000  1.00000
 -7        0.14650          host ceph6
 5    ssd  0.14650              osd.5       up   1.00000  1.00000
```

10. 创建并启用新的 RDB 块池。

```
$ ceph osd pool create rbdpool 32 32
$ ceph osd pool application enable rbdpool rbd
```



注意

命令末尾的数字 32 是分配给这个池的 PG 数量。PG 数量可能会因集群中的 OSD 数量、使用池的预期%的不同而有所不同。您可以使用以下计算器来确定所需的 PG 数量：Ceph Placement Groups(PG)per Pool Calculator 。

11. 验证 RBD 池已创建好。

```
$ ceph osd lspools | grep rbdpool
```

输出示例：

```
 3 rbdpool
```

12. 验证 MDS 服务处于活动状态，并且每个数据中心上具有一个服务。

```
$ ceph orch ps | grep mds
```

输出示例：

```
mds.cephfs.ceph3.cjpbqo    ceph3           running (17m)   117s ago  17m    16.1M      -
16.2.9
mds.cephfs.ceph6.lqmgqt    ceph6           running (17m)   117s ago  17m    16.1M      -
16.2.9
```

13. 创建 CephFS 卷。

```
$ ceph fs volume create cephfs
```

注意

ceph fs volume create 命令还会创建所需的数据和 meta CephFS 池。如需更多信息，请参阅配置和挂载 Ceph 文件系统。

14. 检查 **Ceph** 状态，以验证 MDS 守护进程的部署方式。确保状态为 active，其中 **ceph6** 是这个文件系统的主 MDS，**ceph3** 是次 MDS。

```
$ ceph fs status
```

输出示例：

```
cephfs - 0 clients
======
RANK  STATE         MDS            ACTIVITY     DNS    INOS   DIRS   CAPS
 0    active  cephfs.ceph6.ggjywj  Reqs:    0 /s    10     13     12      0
       POOL          TYPE     USED  AVAIL
cephfs.cephfs.meta  metadata  96.0k   284G
cephfs.cephfs.data    data       0    284G
   STANDBY MDS
cephfs.ceph3.ogcqkl
```

15. 验证 RGW 服务是否处于活动状态。

```
$ ceph orch ps | grep rgw
```

输出示例：

```
rgw.objectgw.ceph3.kkmxgb  ceph3  *:8080       running (7m)      3m ago   7m    52.7M      -
16.2.9
rgw.objectgw.ceph6.xmnpah  ceph6  *:8080       running (7m)      3m ago   7m    53.3M      -
16.2.9
```

3.5.3. 配置 Red Hat Ceph Storage 扩展模式

使用 **cephadm** 完全部署了红帽 Ceph 存储集群后，请使用以下步骤来配置扩展集群模式。新的扩展模式旨在处理 2 个站点的情况。

步骤

1. 使用 ceph mon dump 命令检查监视器所使用的当前选择策略。默认情况下，在 ceph 集群中，连接设置为经典。

```
ceph mon dump | grep election_strategy
```

输出示例：

```
dumped monmap epoch 9
election_strategy: 1
```

2. 将 monitor 选举更改为连接性。

```
ceph mon set election_strategy connectivity
```

3. 再次运行前面的 ceph mon dump 命令，以验证 election_strategy 值。

```
$ ceph mon dump | grep election_strategy
```

输出示例：

```
dumped monmap epoch 10
election_strategy: 3
```

要了解有关不同选择策略的更多信息，请参阅配置监控选举策略。

4. 设置所有 Ceph 监视器的位置：

```
ceph mon set_location ceph1 datacenter=DC1
ceph mon set_location ceph2 datacenter=DC1
ceph mon set_location ceph4 datacenter=DC2
ceph mon set_location ceph5 datacenter=DC2
ceph mon set_location ceph7 datacenter=DC3
```

5. 验证每个监控器是否具有正确的位置。

```
$ ceph mon dump
```

输出示例：

```
epoch 17
fsid dd77f050-9afe-11ec-a56c-029f8148ea14
last_changed 2022-03-04T07:17:26.913330+0000
created 2022-03-03T14:33:22.957190+0000
min_mon_release 16 (pacific)
election_strategy: 3
0: [v2:10.0.143.78:3300/0,v1:10.0.143.78:6789/0] mon.ceph1; crush_location
{datacenter=DC1}
1: [v2:10.0.155.185:3300/0,v1:10.0.155.185:6789/0] mon.ceph4; crush_location
{datacenter=DC2}
2: [v2:10.0.139.88:3300/0,v1:10.0.139.88:6789/0] mon.ceph5; crush_location
{datacenter=DC2}
3: [v2:10.0.150.221:3300/0,v1:10.0.150.221:6789/0] mon.ceph7; crush_location
{datacenter=DC3}
4: [v2:10.0.155.35:3300/0,v1:10.0.155.35:6789/0] mon.ceph2; crush_location
{datacenter=DC1}
```

6. 通过安装 **ceph-base** RPM 软件包来创建使用此 OSD 拓扑的 CRUSH 规则，以便使用 **crushtool** 命令：

```
$ dnf -y install ceph-base
```

要了解有关 CRUSH 规则集的更多信息，请参见 Ceph CRUSH 规则集。

7. 从集群获取编译的 CRUSH map：

```
$ ceph osd getcrushmap > /etc/ceph/crushmap.bin
```

8. 解译 CRUSH map，并将其转换为文本文件，以便能编辑它：

```
$ crushtool -d /etc/ceph/crushmap.bin -o /etc/ceph/crushmap.txt
```

9. 编辑文件 **/etc/ceph/crushmap.txt**，将以下规则添加到 CRUSH map。

```
$ vim /etc/ceph/crushmap.txt
```

```
rule stretch_rule {
        id 1
        type replicated
        min_size 1
        max_size 10
        step take default
        step choose firstn 0 type datacenter
        step chooseleaf firstn 2 type host
        step emit
}
# end crush map
```

本例适用于所有 OpenShift Container Platform 集群中的活动应用程序。

注意

规则 **ID** 必须是唯一的。在示例中，我们只有一个带有 id 0 的 crush 规则，因此我们正在使用 id 1。如果您的部署创建了更多规则，则使用下一个可用 ID。

声明的 CRUSH 规则包含以下信息：

- **运行名称**
  - Description: 用于标识规则的唯一名称。
  - Value: **stretch_rule**
- **id**
  - Description：用于标识规则的唯一整数。
  - Value: **1**
- **type**

- Description：描述存储驱动器复制或纠删代码的规则。
- Value: **replicated**

- **min_size**
  - Description: 如果池制作的副本数少于这个数字，CRUSH 不会选择这一规则。
  - 值 ：1
- **max_size**
  - Description: 如果池制作的副本数多于这个数字，CRUSH 不会选择这一规则。
  - 值 ：10
- **步骤需要默认**
  - 描述：获取名为 **default** 的根存储桶，并开始迭代树。
- **步骤选择第 0 类型数据中心**
  - Description：选择数据中心存储桶，并放入它的子树中。
- **step chooseleaf firstn 2 type host**
  - Description：选择给定类型的存储桶数量。在本例中，两个不同的主机位于其在上一级别输入的数据中心。
- **step emit**
  - Description: 输出当前的值并清除堆栈。通常在规则末尾使用，但也可用于从同一规则的不同树中选取。

10. 从文件 **/etc/ceph/crushmap.txt** 中编译新的 CRUSH map，并将其转换为名为 **/etc/ceph/crushmap2.bin** 的二进制文件：

```
$ crushtool -c /etc/ceph/crushmap.txt -o /etc/ceph/crushmap2.bin
```

11. 注入我们创建回集群的新 crushmap：

```
$ ceph osd setcrushmap -i /etc/ceph/crushmap2.bin
```

输出示例：

```
17
```

注意

数字 17 是一个计数器，它将根据您对 crush 映射所做的更改来增加（18,19 等）。

12. 验证创建的扩展规则现已可供使用。

```
ceph osd crush rule ls
```

输出示例：

```
replicated_rule
stretch_rule
```

13. 启用扩展群集模式。

```
$ ceph mon enable_stretch_mode ceph7 stretch_rule datacenter
```

在本例中，**ceph7** 是仲裁节点，**stretch_rule** 是在上一步中创建的 crush 规则，**datacenter** 是分开的存储桶。

14. 验证我们的所有池正在使用我们在 Ceph 集群中创建的 **stretch_rule** CRUSH 规则：

```
$ for pool in $(rados lspools);do echo -n "Pool: ${pool}; ";ceph osd pool get ${pool}
crush_rule;done
```

输出示例：

```
Pool: device_health_metrics; crush_rule: stretch_rule
Pool: cephfs.cephfs.meta; crush_rule: stretch_rule
Pool: cephfs.cephfs.data; crush_rule: stretch_rule
Pool: .rgw.root; crush_rule: stretch_rule
Pool: default.rgw.log; crush_rule: stretch_rule
Pool: default.rgw.control; crush_rule: stretch_rule
Pool: default.rgw.meta; crush_rule: stretch_rule
Pool: rbdpool; crush_rule: stretch_rule
```

这表明正在运行的红帽 Ceph 存储扩展群集，现在具有仲裁模式。

3.6. 在受管集群中安装 OPENSHIFT DATA FOUNDATION

要在两个 OpenShift Container Platform 集群之间配置存储复制，OpenShift Data Foundation Operator 必须首先安装在每个受管集群上，如下所示：

先决条件

- 确保您已满足 OpenShift Data Foundation 外部部署的硬件要求。有关硬件要求的详情，请参阅外部模式要求。

注意

请参阅 OpenShift Data Foundation 部署指南和特定于基础架构的说明（如 AWS、VMware、BM、Azure 等）。

流程

1. 在每个受管集群中，安装和配置最新的 **OpenShift Data Foundation** 集群。
2. 在安装 operator 后，创建一个 StorageSystem，使用选择 **Full deployment** 类型和 **Connect with external storage platform**，其中您的**后端存储类型**为 **Red Hat Ceph Storage**。
   具体步骤请参考以外部模式部署 OpenShift Data foundation。
   - 您在 **ceph-external-cluster-details-exporter.py script** 中需要最少使用以下三个标记：

     --rbd-data-pool-name

使用在为 OpenShift Container Platform 部署 RHCS 时创建的 RBD 池的名称。例如，池的名称可能为 **rbdpool**。

--rgw-endpoint

以 **<ip_address>:<port>** 格式提供端点。它是与您要配置的 OpenShift Container Platform 集群相同的站点中运行的 RGW 守护进程的 RGW IP。

--run-as-user

每个站点使用不同的客户端名称。

如果在 RHCS 部署过程中使用了默认值，则以下标记是**可选的** ：

--cephfs-filesystem-name

使用在为 OpenShift Container Platform 在 RHCS 部署期间创建的 CephFS 文件系统的名称，默认的文件系统名称是 **cephfs**。

--cephfs-data-pool-name

在用于 OpenShift Container Platform 的 RHCS 部署期间创建的 CephFS 数据池名称后，默认的池称为 **cephfs.data**。

--cephfs-metadata-pool-name

在用于 OpenShift Container Platform 的 RHCS 部署期间创建的 CephFS 元数据池的名称后，默认的池名为 **cephfs.meta**。

- 在 bootstrap 节点（ceph1）上运行以下命令，为 datacenter1 和 datacenter2 中的 RGW 端点获取 IP：

```
ceph orch ps | grep rgw.objectgw
```

输出示例：

```
rgw.objectgw.ceph3.mecpzm  ceph3  *:8080       running (5d)     31s ago   7w     204M
-  16.2.7-112.el8cp
rgw.objectgw.ceph6.mecpzm  ceph6  *:8080       running (5d)     31s ago   7w     204M
-  16.2.7-112.el8cp
```

```
host ceph3
host ceph6
```

输出示例：

```
ceph3.example.com has address 10.0.40.24
ceph6.example.com has address 10.0.40.66
```

- 使用为第一个 ocp 受管集群 cluster1 配置的参数，执行 ceph-external-cluster-details-exporter.py。

```
python3 ceph-external-cluster-details-exporter.py --rbd-data-pool-name rbdpool --cephfs-
filesystem-name cephfs --cephfs-data-pool-name cephfs.cephfs.data  --cephfs-metadata-
pool-name cephfs.cephfs.meta --rgw-endpoint 10.0.40.24:8080 --run-as-user
client.odf.cluster1 > ocp-cluster1.json
```

- 使用为 first ocp 受管集群 cluster2 配置的参数，执行 ceph-external-cluster-details-exporter.py

```
python3 ceph-external-cluster-details-exporter.py --rbd-data-pool-name rbdpool --cephfs-filesystem-name cephfs --cephfs-data-pool-name cephfs.cephfs.data  --cephfs-metadata-pool-name cephfs.cephfs.meta --rgw-endpoint 10.0.40.66:8080 --run-as-user client.odf.cluster2 > ocp-cluster2.json
```

- 将 bootstrap 集群 (ceph1) **ocp-cluster1.json** 和 **ocp-cluster2.json** 这两个文件保存到本地机器中。
- 在部署了外部 ODF 的 **cluster1** 上，使用 OCP 控制台上的文件 **ocp-cluster1.json** 的内容。
- 在部署外部 ODF 的 **cluster2** 上，使用 OCP 控制台上的文件 **ocp-cluster2.json** 的内容。

3. 检查设置，然后选择 **Create StorageSystem**。
4. 使用以下命令验证每个受管集群中 OpenShift Data Foundation 部署是否成功：

```
$ oc get storagecluster -n openshift-storage ocs-external-storagecluster -o jsonpath='{.status.phase}{"\n"}'
```

对于 Multicloud 网关(MCG)：

```
$ oc get noobaa -n openshift-storage noobaa -o jsonpath='{.status.phase}{"\n"}'
```

如果状态结果是 **Ready**，用于**主受管集群**和**二级受管集群**上的查询，则继续执行下一步。

注意

在 OpenShift Web 控制台中，进入到 **Installed Operators → OpenShift Data Foundation → Storage → ocs-storagecluster-storagesystem → Resources**，并验证 **StorageCluster** 的 **Status** 是否为 **Ready**，并在其旁边有一个绿色勾号标记。

3.7. 安装 OPENSHIFT DATA FOUNDATION MULTICLUSTER ORCHESTRATOR OPERATOR

OpenShift Data Foundation Multicluster Orchestrator 是一个控制器，从 Hub 集群上的 OpenShift Container Platform OperatorHub 安装。

流程

1. 在 **Hub 集群中**，进入到 **OperatorHub** 并使用关键字过滤器搜索 **ODF Multicluster Orchestrator**。
2. 点 **ODF Multicluster Orchestrator** 标题。
3. 保留所有默认设置，然后点**安装**。
   确保 operator 资源安装在 **openshift-operators** 中，并可用于所有命名空间。

注意

ODF Multicluster Orchestrator 还会在 RHACM hub 集群上作为依赖项安装 **Openshift DR Hub Operator**。

4. 验证 Operator **Pod** 处于 **Running** 状态。**OpenShift DR Hub operator** 也安装在 **openshift-operators** 命名空间中。

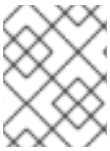
```
$ oc get pods -n openshift-operators
```

输出示例：

```
NAME                                  READY   UP-TO-DATE   AVAILABLE   AGE
odf-multicluster-console-6845b795b9-blxrn   1/1     Running      0          4d20h
odfmo-controller-manager-f9d9dfb59-jbrsd    1/1     Running      0          4d20h
ramen-hub-operator-6fb887f885-fss4w         2/2     Running      0          4d20h
```

3.8. 在集群间配置 SSL 访问

在 **primary 和 secondary 集群**间配置网络 SSL 访问，因此元数据可以以一个安全的方式保持在不同的集群中的 Multicloud Gateway (MCG) **对象存储桶**，以及保持在 **Hub 集群**中验证对对象存储桶的访问。



注意

如果所有 OpenShift 集群都为您的环境使用签名的有效证书集进行部署，则可以跳过本节。

步骤

1. 提取主受管集群的入口证书，并将输出保存到 **primary.crt**。

```
$ oc get cm default-ingress-cert -n openshift-config-managed -o jsonpath="{['data']['ca-bundle\.crt']}" > primary.crt
```

2. 提取二级受管集群的入口证书，并将输出保存到 **secondary.crt**。

```
$ oc get cm default-ingress-cert -n openshift-config-managed -o jsonpath="{['data']['ca-bundle\.crt']}" > secondary.crt
```

3. 创建新的 **ConfigMap** 文件，以使用文件名 **cm-clusters-crt.yaml** 来保存远程集群的证书捆绑包。

注意

如本示例文件所示，每个集群可能有超过三个证书。另外，请确保在从之前创建的 **primary.crt** 和 **secondary.crt** 文件中复制并粘贴后，证书内容会被正确缩进。

```
apiVersion: v1
data:
  ca-bundle.crt: |
    -----BEGIN CERTIFICATE-----
    <copy contents of cert1 from primary.crt here>
    -----END CERTIFICATE-----

    -----BEGIN CERTIFICATE-----
    <copy contents of cert2 from primary.crt here>
    -----END CERTIFICATE-----
```

```
    -----BEGIN CERTIFICATE-----
    <copy contents of cert3 primary.crt here>
    -----END CERTIFICATE----

    -----BEGIN CERTIFICATE-----
    <copy contents of cert1 from secondary.crt here>
    -----END CERTIFICATE-----

    -----BEGIN CERTIFICATE-----
    <copy contents of cert2 from secondary.crt here>
    -----END CERTIFICATE-----

    -----BEGIN CERTIFICATE-----
    <copy contents of cert3 from secondary.crt here>
    -----END CERTIFICATE-----
kind: ConfigMap
metadata:
  name: user-ca-bundle
  namespace: openshift-config
```

4. 在 **Primary 受管集群**, **Secondary 受管集群**, 和 **Hub 集群** 中创建 **ConfigMap**。

```
$ oc create -f cm-clusters-crt.yaml
```

输出示例：

```
configmap/user-ca-bundle created
```

5. 对**主受管集群、二级受管集群**和 **Hub 集群**上的默认代理资源进行补丁。

```
$ oc patch proxy cluster --type=merge  --patch='{"spec":{"trustedCA":{"name":"user-ca-bundle"}}}'
```

输出示例：

```
proxy.config.openshift.io/cluster patched
```

3.9. 在 HUB 集群上创建灾难恢复策略

OpenShift 灾难恢复策略(DRPolicy)资源指定参与灾难恢复解决方案和所需的复制间隔的 OpenShift Container Platform 集群。DRPolicy 是一个集群范围的资源，用户可以应用到需要灾难恢复解决方案的应用程序。

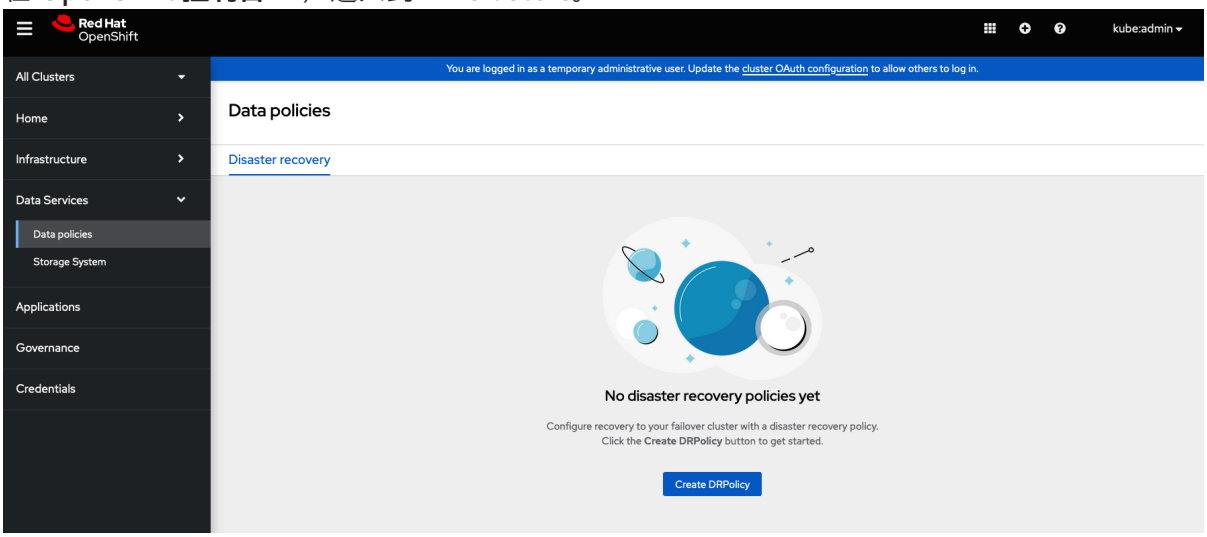
ODF MultiCluster Orchestrator Operator 通过 **Multicluster Web console** 促进每个 **DRPolicy** 和相应的 **DRClusters** 的创建。

先决条件

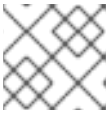
- 确保至少一组两个受管集群。

流程

1. 在 **OpenShift 控制台** 上，进入到 **All Clusters**。



2. 进入 **Data Services** 并点 **Data policies**。
3. 点 **Create DRPolicy**。
4. 输入 **Policy name**。确保每个 DRPolicy 都有一个唯一名称（例如： **ocp4perf1-ocp4perf2**）。
5. 从与此新策略关联的受管集群列表中选择两个集群。
6. **复制策略**根据所选的 OpenShift 集群自动设置为 **sync**。
7. 点 **Create**。
8. 验证 **DRPolicy** 是否已成功创建。对创建的每个 DRPolicy 资源在 **Hub 集群**中运行此命令。



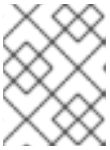
注意

将 <drpolicy_name> 替换为您的唯一名称。

```
$ oc get drpolicy <drpolicy_name> -o jsonpath='{.status.conditions[].reason}{"\n"}'
```

输出示例：

```
Succeeded
```



注意

创建 DRPolicy 时，也会创建两个 DRCluster 资源。验证所有三个资源并且状态显示为 **Succeeded** 最多需要 10 分钟。

9. 验证对象存储桶可以从 **Hub 集群**、**Primary 受管集群**和 **Secondary 受管集群**访问。
   a. 获取 Hub 集群上的 **DRClusters** 的名称。

```
$ oc get drclusters
```

输出示例：

```
NAME       AGE
ocp4perf1   4m42s
ocp4perf2   4m42s
```

b. 使用这个 **DRCluster** 验证命令，检查 S3 可以访问每个受管集群中创建的每个存储桶。

注意

将 <drcluster_name> 替换为您的唯一名称。

```
$ oc get drcluster <drcluster_name> -o jsonpath='{.status.conditions[2].reason}{"\n"}'
```

输出示例：

```
Succeeded
```

注意

确保为在 **Hub 集群**中的两个 **DRClusters** 运行命令。

10. 验证 **OpenShift DR Cluster operator** 已成功在 **Primary 受管集群**和 **Secondary 受管集群**上成功安装。

```
$ oc get csv,pod -n openshift-dr-system
```

输出示例：

```
NAME                                                         DISPLAY                     VERSION
REPLACES   PHASE
clusterserviceversion.operators.coreos.com/odr-cluster-operator.v4.11.0   Openshift DR
Cluster Operator   4.11.0              Succeeded

NAME                                      READY   STATUS    RESTARTS   AGE
pod/ramen-dr-cluster-operator-5564f9d669-f6lbc   2/2     Running   0          5m32s
```

您还可以验证 **OpenShift DR Cluster Operator** 是否在每个受管集群的 **OperatorHub** 上成功安装。

3.10. 为隔离自动化配置 DRCLUSTERS

在应用程序故障切换前启用隔离需要此配置。为了防止在集群中写入持久性卷（由灾难达到的，OpenShift DR 指示 Red Hat Ceph Storage (RHCS)指示 Red Hat Ceph Storage (RHCS)从 RHCS 外部存储隔离节点。本节介绍了如何为 DRCluster 节点添加 IP 或 IP 地址范围。

3.10.1. 将节点 IP 地址添加到 DRClusters

1. 通过在 **Primary 受管集群** 和 **secondary 受管集群**中运行此命令，在受管集群中查找所有 OpenShift 节点的 **IP 地址**。

```
$ oc get nodes -o jsonpath='{range .items[*]}{.status.addresses[?
(@.type=="ExternalIP")].address}{"\n"}{end}'
```

输出示例：

```
10.70.56.118
10.70.56.193
10.70.56.154
10.70.56.242
10.70.56.136
10.70.56.99
```

获得 **IP 地址**后，可以为每个受管集群修改 **DRCluster** 资源。

2. 在 Hub 集群上查找 **DRCluster** 名称。

```
$ oc get drcluster
```

输出示例：

```
NAME        AGE
ocp4perf1   5m35s
ocp4perf2   5m35s
```

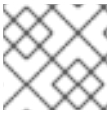
3. 编辑每个 **DRCluster**，在使用您的唯一名称替换 **<drcluster_name>** 后添加您的唯一的 IP地址。

```
$ oc edit drcluster <drcluster_name>
```

```
apiVersion: ramendr.openshift.io/v1alpha1
kind: DRCluster
metadata:
[...]
spec:
  s3ProfileName: s3profile-<drcluster_name>-ocs-external-storagecluster
  ## Add this section
  cidrs:
    - <IP_Address1>/32
    - <IP_Address2>/32
    - <IP_Address3>/32
    - <IP_Address4>/32
    - <IP_Address5>/32
    - <IP_Address6>/32
[...]
```

输出示例：

```
drcluster.ramendr.openshift.io/ocp4perf1 edited
```



注意

有超过 6 个 IP 地址。

在对等 **DRCluster** 资源（如 ocp4perf2）中，为 **Secondary 受管集群**上的 **IP 地址** 修改此 DRCluster 配置。

3.10.2. 在 DRClusters 中添加隔离注解

在所有 DRCluster 资源中添加以下注解。这些注解包括以下说明（测试应用程序故障切换）创建的 **NetworkFence** 资源所需的详细信息。

注意

将 *<drcluster_name>* 替换为您的唯一名称。

```
$ oc edit drcluster <drcluster_name>
```

```
apiVersion: ramendr.openshift.io/v1alpha1
kind: DRCluster
metadata:
  ## Add this section
  annotations:
    drcluster.ramendr.openshift.io/storage-clusterid: openshift-storage
    drcluster.ramendr.openshift.io/storage-driver: openshift-storage.rbd.csi.ceph.com
    drcluster.ramendr.openshift.io/storage-secret-name: rook-csi-rbd-provisioner
    drcluster.ramendr.openshift.io/storage-secret-namespace: openshift-storage
[...]
```

输出示例：

```
drcluster.ramendr.openshift.io/ocp4perf1 edited
```

确保为 **DRCluster** 资源添加这些注解（例如：**ocp4perf1** 和 **ocp4perf2**）。

3.11. 为测试灾难恢复解决方案创建示例应用程序

OpenShift Data Foundation 灾难恢复 (DR) 解决方案为由 RHACM 管理的应用程序支持灾难恢复。如需了解更多详细信息，请参阅管理应用程序。

注意

OpenShift Data Foundation DR 解决方案不支持 ApplicationSet，这是通过 ArgoCD 部署的应用程序所需的。

当应用程序在 DRPolicy 中移动应用程序以便进行故障转移或重新定位要求时，该解决方案使用 PlacementRule 编排 RHACM 应用程序放置。以下小节详细介绍了如何对应用程序应用 **DRPolicy**，以及如果在集群不可用期间或之后管理应用程序放置生命周期。

注意

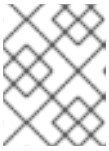
没有 cluster-admin 权限的 OpenShift 用户，请参阅知识库文章 如何为应用程序用户分配必要的权限来执行灾难恢复操作。

3.11.1. 创建示例应用程序

为了测试从 **Primary 受管集群**到 **Secondary 受管集群**的 **failover** 和 **relocate**，我们需要一个简单的应用程序。

先决条件

- 在创建用于常规消耗的应用程序时，请确保应用程序仅部署到一个集群。
- 使用名为 **busybox** 的示例应用作为示例。
- 当应用程序通过或重新定位失败时，确保使用全局流量管理器(GTM)或 Global Server Load Balancing (GLSB)服务配置应用程序的所有外部路由。
- 作为最佳实践，将属于 Red Hat Advanced Cluster Management (RHACM) 订阅的 Red Hat Advanced Cluster Management (RHACM) 订阅分组在一起，以指代单个 Placement Rule 到 DR 保护它们作为组。进一步将它们创建为一个应用程序，用于订阅的逻辑分组，以备将来 DR 操作，如故障转移和重新定位。

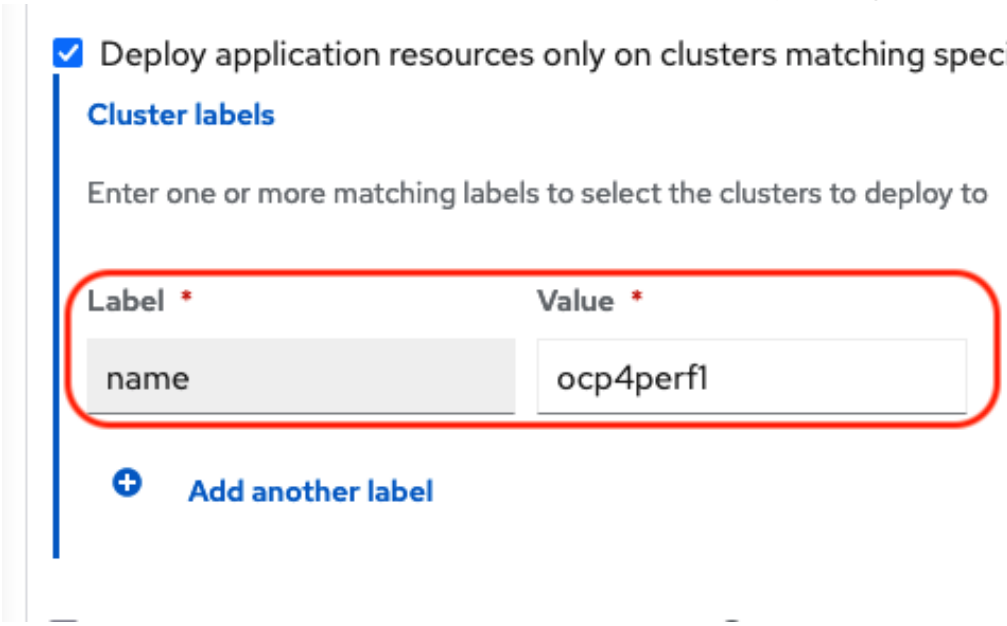


注意

如果不相关的订阅引用与放置操作相同的放置规则，它们也会受到 DR 保护，因为 DR 工作流控制引用放置规则的所有订阅。

流程

1. 在 Hub 集群中，进入到 **Applications**，再点 **Create application**。
2. 选择类型 **Subscription**。
3. 输入应用程序 **Name** （如 **busybox**）和 **Namespace** （如 **busybox-sample**）。
4. 在 Repository location for resources 部分中，选择 **Repository type Git**。
5. 输入示例应用程序的 Git 存储库 URL、github **Branch** 和 **Path**，在其中创建资源 **busybox** Pod 和 PVC。
   使用示例应用程序存储库作为 **https://github.com/red-hat-storage/ocm-ramen-samples**，其中 **Branch** 是 **release-4.12**，**Path** 是 **busybox-odr-metro**。
6. 滚动表单，直到您看到 **Deploy application resources only on clusters matching specified labels**，然后在 **RHACM** 集群列表视图中添加一个值为 **Primary managed cluster** 名称的标签。



7. 点位于右上角的 **Create**。

在接下来的屏幕中，进入 **Topology** 选项卡。您可以看到应用程序拓扑上有所有的绿色对勾图标。

注意

要获取更多信息，请点击任何拓扑元素，拓扑视图右侧会显示一个窗口。

8. 验证示例应用程序部署。
现在，**busybox** 应用程序已部署到首选集群，现在可以验证部署。

登录到您的受管集群，其中 **busybox** 由 RHACM 部署。

```
$ oc get pods,pvc -n busybox-sample
```

输出示例：

```
NAME                      READY   STATUS    RESTARTS   AGE
pod/busybox-67bf494b9-zl5tr   1/1     Running   0          77s


NAME                                 STATUS   VOLUME                                     CAPACITY   ACCESS
MODES   STORAGECLASS              AGE
persistentvolumeclaim/busybox-pvc    Bound    pvc-c732e5fe-daaf-4c4d-99dd-462e04c18412
5Gi        RWO            ocs-storagecluster-ceph-rbd   77s
```

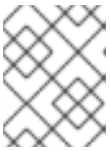
3.11.2. 将 DRPolicy 应用到示例应用程序

先决条件

- 确保 DRPolicy 中引用的两个受管集群都可以访问。如果不能访问，则应用程序在两个集群都在线前无法受到 DR 保护。

流程

1. 在 Hub 集群中，返回到 Multicluster Web 控制台，进入到 **All Clusters**。
2. 登录到 **All Clusters** 下列出的所有集群。
3. 进入到 **Data Services**，然后点 **Data policies**。
4. 点 DRPolicy 末尾的 Actions 菜单，以查看可用操作的列表。
5. 点 **Apply DRPolicy**。
6. 显示 **Apply DRPolicy** 模式时，选择 **busybox** 应用程序，再输入 **PVC 标签** 作为 **appname=busybox**。



注意

如果选择了同一应用程序或多个应用程序下的多个放置规则，则应用程序命名空间中的所有 PVC 都会被默认保护。

7. 点**应用**。

8. 验证在 Hub 集群的 **busybox-sample** 命名空间中创建了 **DRPlacementControl** 或 **DRPC**，并它的 **CURRENTSTATE** 显示为 **Deployed**。此资源用于此应用的故障切换和重定位操作。

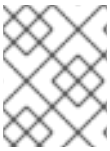
```
$ oc get drpc -n busybox-sample
```

输出示例：

```
NAME                    AGE    PREFERREDCLUSTER   FAILOVERCLUSTER
DESIREDSTATE   CURRENTSTATE
busybox-placement-1-drpc   6m59s   ocp4perf1                                  Deployed
```

3.11.3. 删除示例应用程序

您可以使用 RHACM 控制台删除示例 application **busybox**。



注意

在故障转移和故障恢复（重复）测试完成以及应用程序已可以从 RHACM 和受管集群中删除之前，不应执行示例应用程序的说明。

流程

1. 在 RHACM 控制台上，前往 **Applications**。
2. 搜索要删除的示例应用程序（例如 **busybox**）。
3. 点击您要删除的应用程序旁边的 Action Menu (⋮)。
4. 点击 **Delete application**。
   选择了 **Delete application** 时，将显示一个新屏幕，询问是否还应删除与应用相关的资源。
5. 选中 **Remove application 相关资源**复选框，以删除 Subscription 和 PlacementRule。
6. 单击 **Delete**。这将删除主受管集群（或应用程序所运行的任何群集）上的 busybox 应用程序。
7. 除了使用 RHACM 控制台删除资源外，在删除 **busybox** 应用后，还必须立即删除 **DRPlacementControl**。
   a. 登录到 Hub 集群的 OpenShift Web 控制台，再进入为 **busybox-sample** 项目安装的 Operators。
   b. 单击 **OpenShift DR Hub Operator**，然后单击 **DRPlacementControl** 选项卡。
   c. 点击您要删除的 **busybox** 应用程序 DRPlacementControl 旁边的 Action Menu（⋮）。
   d. 单击 **Delete DRPlacementControl**。
   e. 单击 **Delete**。

注意

此过程可用于使用 **DRPlacementControl** 资源删除任何应用。

3.12. 受管集群之间的应用程序故障切换

当受管集群因任何原因而不可用时，执行故障转移。这种故障转移方法是基于应用。

先决条件

- 如果您的设置具有主动和被动 RHACM hub 集群，请参阅使用 Advanced Cluster Management Hub 恢复。
- 当主集群处于 **Ready** 以外的状态时，请检查集群的实际状态，因为它可能需要一些时间才能更新。
  1. 导航到 **RHACM 控制台 → Infrastructure → Clusters → Cluster list**选项卡。
  2. 在执行故障转移操作前，请检查两个受管集群的状态。
     但是，当您故障转移的集群处于 *Ready* 状态时，仍可执行故障转移操作。
- 为了故障转移运行所有应用的 OpenShift 集群，必须隔离所有应用，以便与外部 OpenShift Data Foundation 外部存储集群进行通信。这可以防止两个受管集群同时写入同一持久性卷。要 **Fence** 的 OpenShift 集群是应用当前运行的位置。

流程

1. 在 **Hub 集群**上启用隔离。
   a. 打开 CLI 终端，再编辑 **DRCluster 资源**。

   小心

   在隔离了受管集群后，**所有** 从应用程序到 OpenShift Data Foundation 外部存储集群的通信将失败，一些 **Pod** 都将处于不健康状态（例如：**CreateContainerError**,**CrashLoopBackOff**）现已被隔离的集群。

   注意

   将 *<drcluster_name>* 替换为您的唯一名称。

   ```
   $ oc edit drcluster <drcluster_name>
   ```

   ```
   apiVersion: ramendr.openshift.io/v1alpha1
   kind: DRCluster
   metadata:
   [...]
   spec:
     ## Add this line
     clusterFence: Fenced
     cidrs:
     [...]
   [...]
   ```

   输出示例：

   ```
   drcluster.ramendr.openshift.io/ocp4perf1 edited
   ```

   b. 为 **Primary 受管集群** 验证 **Hub 集群**中的隔离状态。

注意

将 *<drcluster_name>* 替换为您的唯一名称。

```
$ oc get drcluster.ramendr.openshift.io <drcluster_name> -o jsonpath='{.status.phase}
{"\n"}'
```

输出示例：

```
Fenced
```

c. 验证属于 OpenShift Container Platform 集群节点的 IP 现在是否在 blocklist 中。

```
$ ceph osd blocklist ls
```

输出示例

```
cidr:10.1.161.1:0/32 2028-10-30T22:30:03.585634+0000
cidr:10.1.161.14:0/32 2028-10-30T22:30:02.483561+0000
cidr:10.1.161.51:0/32 2028-10-30T22:30:01.272267+0000
cidr:10.1.161.63:0/32 2028-10-30T22:30:05.099655+0000
cidr:10.1.161.129:0/32 2028-10-30T22:29:58.335390+0000
cidr:10.1.161.130:0/32 2028-10-30T22:29:59.861518+0000
```

2. 在 Hub 集群中，进入到 **Applications**。
3. 点应用程序行末尾的 **Actions** 菜单，以查看可用操作的列表。
4. 点 **Failover application**。
5. 显示 **Failover 应用程序** 弹出窗口中，**选择策略**和**目标集群**，相关的应用程序将在出现灾难时故障转移。
6. 默认情况下，选择通过 **复制**应用程序资源的订阅组。**点选择订阅组**下拉菜单，验证默认选择**或修改此**设置。
7. 检查 **Failover readiness** 的状态。
   - 如果状态是 **Ready** 且带有一个绿色勾号，这表示目标集群已就绪，可启动故障转移。继续执行第 7 步。
   - 如果状态是 **Unknown** 或 **Not ready**，请等待到状态变为 **Ready**。
8. 点 **Initiate**。busybox 资源现在在目标集群上创建。
9. 关闭模态窗口，并使用 Applications 页面中的 **Data policies** 列跟踪状态。
10. 验证活动状态是否为应用的 **FailedOver**。
    a. 进入 **Applications → Overview** 选项卡。
    b. 在 **Data policy** 列中，点您要将策略应用到的应用程序的**策略**链接。
    c. 在 **Data policies** 模态页面中，点 **View more details** 链接。

3.13. 在受管集群间重新定位应用程序

当所有受管集群都可用时，将应用程序重新定位到首选位置。

前提条件

- 如果您的设置具有主动和被动 RHACM hub 集群，请参阅使用 Advanced Cluster Management Hub 恢复。
- 当主集群处于 **Ready** 以外的状态时，请检查集群的实际状态，因为它可能需要一些时间才能更新。只有在主集群和首选集群启动并运行时才可执行重新定位。
  1. 导航到 **RHACM 控制台 → Infrastructure → Clusters → Cluster list**选项卡。
  2. 在执行重新定位操作前，请单独检查两个受管集群的状态。
- 在取消隔离前，验证应用程序是否已从集群中清理。

流程

1. 在 Hub 集群中禁用隔离功能。
   a. 编辑此集群的 **DRCluster 资源**。

注意

将 *<drcluster_name>* 替换为您的唯一名称。

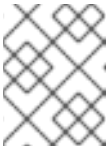
```
$ oc edit drcluster <drcluster_name>
```

```
apiVersion: ramendr.openshift.io/v1alpha1
kind: DRCluster
metadata:
[...]
spec:
  cidrs:
  [...]
  ## Modify this line
  clusterFence: Unfenced
  [...]
[...]
```

输出示例：

```
drcluster.ramendr.openshift.io/ocp4perf1 edited
```

   b. 正常重新引导了 **Fenced** 的 OpenShift Container Platform 节点。重启后，需要重启来恢复 I/O 操作，以避免进一步恢复编配失败。按照安全重启节点的步骤，重新引导集群中的所有节点。



注意

先确保所有节点已被封锁和排空，然后再重新引导并在节点上执行 uncordon 操作。

c. 在所有 OpenShift 节点被重新引导并处于 **Ready** 状态后，通过在主受管集群中运行此命令（或任何集群是 Unfenced），验证所有 Pod 都处于健康状态。

```
oc get pods -A | egrep -v 'Running|Completed'
```

输出示例：

```
NAMESPACE                                 NAME
READY   STATUS      RESTARTS      AGE
```

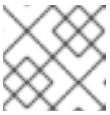
此查询的输出应该为零个 Pod，然后继续下一步。

重要

如果因为严重的存储通信导致 pod 仍然处于不健康状态，请在继续操作前进行故障排除并解决。由于存储集群位于 OpenShift 的外部，因此在 OpenShift 应用正常运行的站点中断后，还必须正确恢复它。

另外，您可以使用 OpenShift Web 控制台仪表板和 Overview 选项卡来评估应用程序和外部 ODF 存储集群的健康状态。详细的 OpenShift Data Foundation 仪表板可通过 **Storage → Data Foundation** 找到。

d. 验证 **Unfenced** 集群是否处于健康状态。为 Primary 受管集群验证 Hub 集群中的隔离状态。



注意

将 *<drcluster_name>* 替换为您的唯一名称。

```
$ oc get drcluster.ramendr.openshift.io <drcluster_name> -o jsonpath='{.status.phase}
{"\n"}'
```

输出示例：

```
Unfenced
```

e. 验证属于 OpenShift Container Platform 集群节点的 IP 不在 blocklist 中。

```
$ ceph osd blocklist ls
```

确保您没有看到隔离过程中添加的 IP。

2. 在 Hub 集群中，进入到 **Applications**。
3. 点应用程序行末尾的 **Actions** 菜单，以查看可用操作的列表。
4. 点 **Relocate application**。

5. 当 **Relocate application** 弹出窗口显示时，选择 **policy** 和 **target cluster**，在出现灾难时相关的应用程序将重新定位到其中。

6. 默认情况下，选择部署应用程序资源的订阅组。点选择**订阅组**下拉菜单，验证默认选择或修改此设置。

7. 检查 **Relocation readiness** 的状态。

   - 如果状态是 **Ready** 且带有一个绿色勾号，这表示目标集群已准备好重定位来进行启动。继续执行第 7 步。

   - 如果状态是 **Unknown** 或 **Not ready**，请等待到状态变为 **Ready**。

8. 点 **Initiate**。busybox 资源现在在目标集群上创建。

9. 关闭模态窗口，并使用 Applications 页面中的 **Data policies** 列跟踪状态。

10. 验证应用程序的活动状态是否显示为 **Relocated**。

    a. 进入 **Applications → Overview** 选项卡。

    b. 在 **Data policy** 列中，点您要将策略应用到的应用程序的**策略**链接。

    c. 在 **Data policies** 模态页面中，点 **View more details** 链接。

第 4 章 用于 OPENSHIFT DATA FOUNDATION 的 REGION-DR 解决方案 [技术预览]

重要

使用 Advanced Cluster Management 为 Regional-DR 配置 OpenShift 数据基础是一项技术预览功能，并受技术预览支持限制。技术预览功能不受红帽产品服务等级协议（SLA）支持，且功能可能并不完整。红帽不推荐在生产环境中使用它们。这些技术预览功能可以使用户提早试用新的功能，并有机会在开发阶段提供反馈意见。

如需更多信息，请参阅技术预览功能支持范围。

4.1. 区域 DR 解决方案的组件

Region-DR 由 Red Hat Advanced Cluster Management for Kubernetes 和 OpenShift Data Foundation 组件组成，以便在 Red Hat OpenShift Container Platform 集群中提供应用程序和数据移动性。

Red Hat Advanced Cluster Management for Kubernetes

Red Hat Advanced Cluster Management (RHACM)提供了管理多个集群和应用程序生命周期的功能。因此，它充当多集群环境中的控制平面。

RHACM 分为两个部分：

- RHACM Hub：包括在多集群 control plane 上运行的组件
- 受管集群：包括在受管理的集群中运行的组件

有关此产品的更多信息，请参阅 RHACM 文档 和 RHACM"管理应用程序"文档。

OpenShift Data Foundation

OpenShift Data Foundation 为 OpenShift Container Platform 集群中有状态应用程序提供部署和管理存储的功能。

OpenShift Data Foundation 由 Ceph 作为存储提供商提供支持，其生命周期由 OpenShift Data Foundation 组件堆栈中的 Rook 进行管理。Ceph-CSI 为有状态应用提供持久卷的调配与管理。

OpenShift Data Foundation 堆栈现在增强了以下灾难恢复功能：

- 启用 RBD 块池，以便在 OpenShift Data Foundation 实例（集群）之间进行镜像。
- 可以在一个 RBD 块池中 mirror 特定的镜像
- 提供 csi-addons 以管理每个持久性卷声明(PVC)镜像

OpenShift DR

OpenShift DR 是跨一组使用 RHACM 部署和管理的有状态应用程序的灾难恢复编排器，并提供云原生接口来编排应用程序状态在持久性卷上的生命周期。它们是：

- 保护跨 OpenShift 集群的应用程序及其状态关系
- 将一个应用程序及其状态转移到一个对等集群
- 将一个应用程序及其状态重新定位到之前部署的集群

OpenShift DR 被分成三个组件：

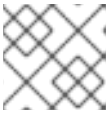
- **ODF Multicluster Orchestrator**: 安装在多集群控制平面 (RHACM Hub) 中，它会编配配置并针对 Metro 和 Regional D 关系对等 OpenShift Data Foundation 集群
- **OpenShift DR Hub Operator**：在 hub 集群上作为 ODF 多集群安装的一部分自动安装，以编配启用 DR 应用程序的故障转移或重新定位。
- **OpenShift DR Cluster Operator**：在属于 Metro 和 Regional DR relationship的每个受管集群中自动安装，用于管理应用程序的所有 PVC 的生命周期。

4.2. 区域DR 部署工作流

本节概述了在两个不同的 OpenShift Container Platform 集群中使用 OpenShift Data Foundation 的最新版本 配置和部署区域 DR 功能所需的步骤。除了两个受管集群外，还需要第三个 OpenShift Container Platform 集群来部署 Red Hat Advanced Cluster Management (RHACM)。

要配置基础架构，请按照给定的顺序执行以下步骤：

1. 确保满足作为 DR 解决方案的三个集群（Hub、Primary 和 Secondary Openshift Container Platform）集群的要求。请参阅启用区域DR 的要求。
2. 安装 OpenShift Data Foundation operator，并在 Primary 和 Secondary 受管集群上创建存储系统。请参阅在受管集群上创建 OpenShift Data Foundation 集群。
3. 在 Hub 集群上安装 ODF Multicluster Orchestrator。请参阅在 Hub 集群上安装 ODF 多集群编排器。
4. 配置 Hub、Primary 和 Secondary 集群之间的 SSL 访问。请参阅配置跨集群的 SSL 访问。
5. 创建 DRPolicy 资源，以用于在跨 Primary 和 Secondary 应用程序所需的 DR 保护的应用程序。请参阅在 Hub 集群上创建灾难恢复策略。



注意

可以有多个策略。

测试您的灾难恢复解决方案：

- 使用 RHACM 控制台创建示例应用程序。请参阅创建示例应用程序。
- 在受管集群之间使用示例应用程序测试故障切换和重定位操作。请参阅应用程序故障切换和重新定位应用程序。

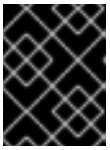
4.3. 启用区域DR 的要求

Red Hat OpenShift Data Foundation 支持的灾难恢复功能需要满足以下所有先决条件，才能成功实施灾难恢复解决方案：

- 您必须有三个在它们之间具有网络可访问性的 OpenShift 集群：
  - **Hub 集群**，Red Hat Advanced Cluster Management for Kubernetes (RHACM operator) 在其中安装。
  - **Primary 受管集群**，OpenShift Data Foundation 在其中安装。

- **Secondary 受管集群**，OpenShift Data Foundation 在其中安装。

- 确保在 Hub 集群中安装 RHACM 操作符和 MultiClusterHub。具体步骤请查看 RHACM 安装指南。
  成功安装 Operator 后，用户界面中会显示一个带有 Web 控制台更新可用消息的弹出窗口。点此弹出窗口中的 **Refresh Web 控制台**来反映控制台更改。



重要

用户负责确保正确配置应用流量路由和重定向。目前不支持对应用程序流量路由的配置和更新。

- 在 Hub 集群中，进入到 **All Clusters → Infrastructure → Clusters**。确保您已使用 RHACM 控制台导入或创建了 **主要受管集群**和**次要受管集群**。具体步骤请参阅创建集群和将目标受管集群导入到 hub 集群。

- 受管集群必须具有非重叠的网络。
  要使用 Submariner 附加组件连接受管 OpenShift 集群和服务网络，您需要对每个受管集群运行以下命令来验证两个集群是否具有非覆盖网络。

```
$ oc get networks.config.openshift.io cluster -o json | jq .spec
```

Primary 集群的示例输出：

```
{
  "clusterNetwork": [
    {
      "cidr": "10.5.0.0/16",
      "hostPrefix": 23
    }
  ],
  "externalIP": {
    "policy": {}
  },
  "networkType": "OpenShiftSDN",
  "serviceNetwork": [
    "10.15.0.0/16"
  ]
}
```

Secondary 集群的输出示例：

```
{
  "clusterNetwork": [
    {
      "cidr": "10.6.0.0/16",
      "hostPrefix": 23
    }
  ],
  "externalIP": {
    "policy": {}
  },
  "networkType": "OpenShiftSDN",
  "serviceNetwork": [
```

```
    "10.16.0.0/16"
  ]
}
```

如需更多信息，请参阅 Submariner 附加组件文档。

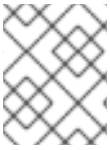
- 确保受管集群可以使用 **Submariner 附加组件**进行连接。在识别并确保集群和服务网络具有非覆盖范围后，使用 RHACM 控制台和**集群集**为每个受管集群安装 **Submariner 附加组件**。具体步骤请参阅 Submariner 文档。

小心

不要选择 **Enable Globalnet**，因为受管集群会有重叠的集群和服务网络。目前，区域灾难恢复不支持使用 **Globalnet**。在继续操作前，请确保集群和服务网络没有重叠。

4.4. 在受管集群上创建 OPENSHIFT DATA FOUNDATION 集群。

要在两个 OpenShift Container Platform 集群间配置存储复制，请在安装 OpenShift Data Foundation Operator 后创建一个 OpenShift Data Foundation 存储系统。



注意

请参阅 OpenShift Data Foundation 部署指南和特定于基础架构的说明（如 AWS、VMware、BM、Azure 等）。

流程

1. 在每个受管集群中，安装和配置最新的 **OpenShift Data Foundation** 集群。
   如需有关 OpenShift Data Foundation 部署的信息，请参阅基础架构特定的部署指南 （如 AWS、VMware、Bare metal、Azure）。
2. 使用以下命令验证每个受管集群中 OpenShift Data Foundation 部署是否成功：

```
$ oc get storagecluster -n openshift-storage ocs-storagecluster -o jsonpath='{.status.phase}{"\n"}'
```

对于 Multicloud 网关(MCG)：

```
$ oc get noobaa -n openshift-storage noobaa -o jsonpath='{.status.phase}{"\n"}'
```

如果状态结果是 **Ready**，用于**主受管集群**和**二级受管集群**上的查询，则继续执行下一步。

注意

在 OpenShift Web 控制台中，进入到 **Installed Operators → OpenShift Data Foundation → Storage → ocs-storagecluster-storagesystem → Resources**，并验证 **StorageCluster** 的 **Status** 是否为 **Ready**，并在其旁边有一个绿色勾号标记。

4.5. 安装 OPENSHIFT DATA FOUNDATION MULTICLUSTER ORCHESTRATOR OPERATOR

OpenShift Data Foundation Multicluster Orchestrator 是一个控制器，从 Hub 集群上的 OpenShift Container Platform OperatorHub 安装。

流程

1. 在 **Hub 集群中**，进入到 **OperatorHub** 并使用关键字过滤器搜索 **ODF Multicluster Orchestrator**。
2. 点 **ODF Multicluster Orchestrator** 标题。
3. 保留所有默认设置，然后点**安装**。
   确保 operator 资源安装在 **openshift-operators** 中，并可用于所有命名空间。

注意

ODF Multicluster Orchestrator 还会在 RHACM hub 集群上作为依赖项安装 **Openshift DR Hub Operator**。

4. 验证 Operator **Pod** 处于 **Running** 状态。**OpenShift DR Hub operator** 也安装在 **openshift-operators** 命名空间中。

```
$ oc get pods -n openshift-operators
```

输出示例：

```
NAME                                READY   UP-TO-DATE   AVAILABLE   AGE
odf-multicluster-console-6845b795b9-blxrn   1/1     Running     0          4d20h
odfmo-controller-manager-f9d9dfb59-jbrsd    1/1     Running     0          4d20h
ramen-hub-operator-6fb887f885-fss4w         2/2     Running     0          4d20h
```

4.6. 在集群间配置 SSL 访问

在 **primary 和 secondary 集群**间配置网络 SSL 访问，因此元数据可以以一个安全的方式保持在不同的集群中的 Multicloud Gateway (MCG) **对象存储桶**，以及保持在 **Hub 集群**中验证对对象存储桶的访问。

注意

如果所有 OpenShift 集群都为您的环境使用签名的有效证书集进行部署，则可以跳过本节。

步骤

1. 提取主受管集群的入口证书，并将输出保存到 **primary.crt**。

```
$ oc get cm default-ingress-cert -n openshift-config-managed -o jsonpath="{['data']['ca-bundle\.crt']}" > primary.crt
```

2. 提取二级受管集群的入口证书，并将输出保存到 **secondary.crt**。

```
$ oc get cm default-ingress-cert -n openshift-config-managed -o jsonpath="{['data']['ca-bundle\.crt']}" > secondary.crt
```

3. 创建新的 **ConfigMap** 文件，以使用文件名 **cm-clusters-crt.yaml** 来保存远程集群的证书捆绑包。

注意

如本示例文件所示，每个集群可能有超过三个证书。另外，请确保在从之前创建的 **primary.crt** 和 **secondary.crt** 文件中复制并粘贴后，证书内容会被正确缩进。

```
apiVersion: v1
data:
  ca-bundle.crt: |
    -----BEGIN CERTIFICATE-----
    <copy contents of cert1 from primary.crt here>
    -----END CERTIFICATE-----

    -----BEGIN CERTIFICATE-----
    <copy contents of cert2 from primary.crt here>
    -----END CERTIFICATE-----

    -----BEGIN CERTIFICATE-----
    <copy contents of cert3 primary.crt here>
    -----END CERTIFICATE----

    -----BEGIN CERTIFICATE-----
    <copy contents of cert1 from secondary.crt here>
    -----END CERTIFICATE-----

    -----BEGIN CERTIFICATE-----
    <copy contents of cert2 from secondary.crt here>
    -----END CERTIFICATE-----

    -----BEGIN CERTIFICATE-----
    <copy contents of cert3 from secondary.crt here>
    -----END CERTIFICATE-----
kind: ConfigMap
metadata:
  name: user-ca-bundle
  namespace: openshift-config
```

4. 在 **Primary 受管集群**, **Secondary 受管集群**, 和 **Hub 集群** 中创建 **ConfigMap**。

```
$ oc create -f cm-clusters-crt.yaml
```

输出示例：

```
configmap/user-ca-bundle created
```

5. 对**主受管集群**、**二级受管集群**和 **Hub 集群**上的默认代理资源进行补丁。

```
$ oc patch proxy cluster --type=merge  --patch='{"spec":{"trustedCA":{"name":"user-ca-bundle"}}}'
```

输出示例：

```
proxy.config.openshift.io/cluster patched
```

4.7. 在 HUB 集群上创建灾难恢复策略

OpenShift 灾难恢复策略(DRPolicy)资源指定参与灾难恢复解决方案和所需的复制间隔的 OpenShift Container Platform 集群。DRPolicy 是一个集群范围的资源，用户可以应用到需要灾难恢复解决方案的应用程序。

ODF MultiCluster Orchestrator Operator 通过 **Multicluster Web console** 促进每个 **DRPolicy** 和相应的 **DRClusters** 的创建。

注意

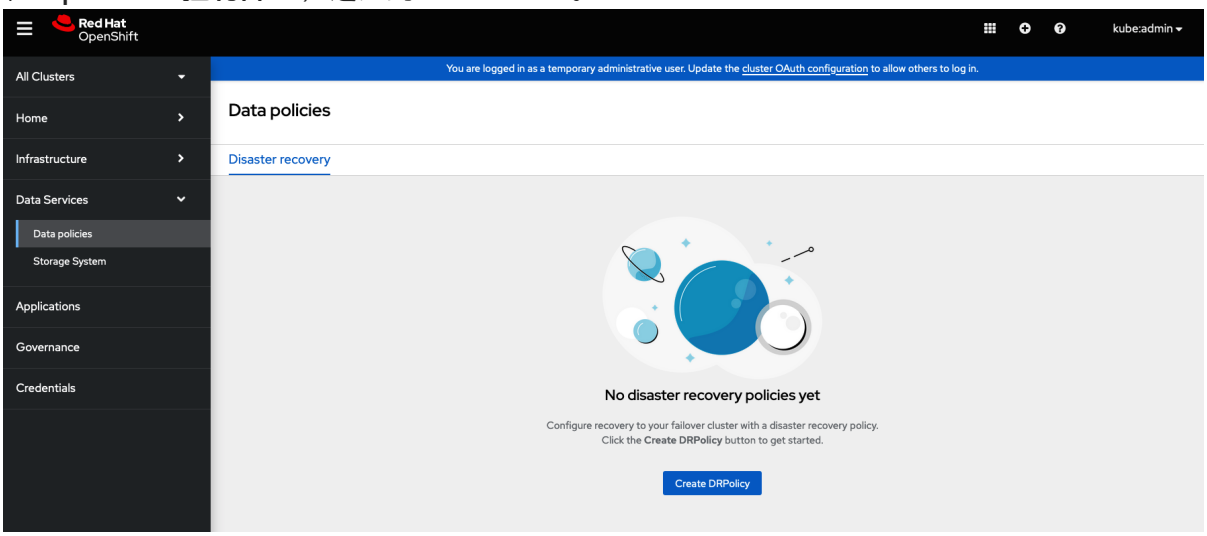
在初始运行时，会自动安装 VolSync operator。VolSync 用于在两个集群之间设置卷复制，以保护基于 CephF 的 PVC。复制功能默认为启用。

先决条件

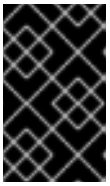
- 确保至少一组两个受管集群。

流程

1. 在 **OpenShift 控制台** 上，进入到 **All Clusters**。



2. 进入 **Data Services** 并点 **Data policies**。
3. 点 **Create DRPolicy**。
4. 输入 **Policy name**。确保每个 DRPolicy 都有一个唯一名称（例如： **ocp4bos1-ocp4bos2-5m**）。
5. 从与此新策略关联的受管集群列表中选择两个集群。
6. **复制策略**根据所选的 OpenShift 集群自动设置为 **Asynchronous**(async)，一个 **Sync schedule** 选项变为可用。
7. 设置**同步调度**。



重要

对于每个需要的复制间隔，必须使用唯一名称（如 **ocp4bos1-ocp4bos2-10m**）创建新的 **DRPolicy**。可以选择同一集群，但 **Sync 调度** 可以用分钟/小时/天内使用不同的复制间隔配置。最小值为一分钟。

8. 点 **Create**。
9. 验证 **DRPolicy** 是否已成功创建。对创建的每个 DRPolicy 资源在 **Hub 集群**中运行此命令。

注意

将 <drpolicy_name> 替换为您的唯一名称。

```
$ oc get drpolicy <drpolicy_name> -o jsonpath='{.status.conditions[].reason}{"\n"}'
```

输出示例：

```
Succeeded
```

注意

创建 DRPolicy 时，也会创建两个 DRCluster 资源。验证所有三个资源并且状态显示为 **Succeeded** 最多需要 10 分钟。

10. 验证对象存储桶可以从 **Hub 集群**、**Primary 受管集群**和 **Secondary 受管集群**访问。
    a. 获取 Hub 集群上的 **DRClusters** 的名称。

```
$ oc get drclusters
```

输出示例：

```
NAME      AGE
ocp4bos1  4m42s
ocp4bos2  4m42s
```

    b. 使用这个 **DRCluster** 验证命令，检查 S3 可以访问每个受管集群中创建的每个存储桶。

注意

将 <drcluster_name> 替换为您的唯一名称。

```
$ oc get drcluster <drcluster_name> -o jsonpath='{.status.conditions[2].reason}{"\n"}'
```

输出示例：

```
Succeeded
```

注意

确保为在 **Hub 集群**中的两个 **DRClusters** 运行命令。

11. 验证 **OpenShift DR Cluster operator** 已成功在 **Primary 受管集群**和 **Secondary 受管集群**上成功安装。

```
$ oc get csv,pod -n openshift-dr-system
```

输出示例：

```
NAME                                                                DISPLAY                       VERSION
REPLACES   PHASE
clusterserviceversion.operators.coreos.com/odr-cluster-operator.v4.11.0   Openshift DR
Cluster Operator   4.11.0               Succeeded

NAME                                       READY   STATUS    RESTARTS   AGE
pod/ramen-dr-cluster-operator-5564f9d669-f6lbc   2/2     Running   0          5m32s
```

您还可以验证 **OpenShift DR Cluster Operator** 是否在每个受管集群的 **OperatorHub** 上成功安装。

12. 验证 ODF 镜像 **Primary 受管集群**和 **Secondary 受管集群**中的 **daemon** 健康的状态。

```
$ oc get cephblockpool ocs-storagecluster-cephblockpool -n openshift-storage -o
jsonpath='{.status.mirroringStatus.summary}{"\n"}'
```

输出示例：

```
{"daemon_health":"OK","health":"OK","image_health":"OK","states":{}}
```

小心

daemon_health 和 **health** 从 **Warning** 变为 **OK** 可能需要最多 10 分钟。如果状态最终未变为 **OK**，则使用 RHACM 控制台验证受管集群之间的 Submariner 连接是否仍然处于健康状态。在所有值都变为 **OK** 之前不要继续。

13. 当使用 **VolSync** 保护基于 CephF 的 PVC 时，请配置 **VolSync** 复制方法。默认复制方法是使用快照。快照在源中进行，并同步到临时目标 PVC。完成同步后，会从这个临时 PVC 中获取另一个快照并保存到目标集群上。在故障切换时，应用程序 PVC 会从集群的最新快照中恢复。
当使用包含数千个文件的 PVC 时，可能需要将快照用作复制方法，因为 CephFS 需要很长时间才能从快照创建可写 PVC。另外，当将复制方法用作快照时，在故障转移或复制后，整个 PVC 必须同步到另一端。这在高延迟网络和大型 PVC 大小上是一个非常昂贵的操作。

为避免这些问题，可以使用"直接"复制方法。这个方法是首选的，因为与应用程序 PVC 直接进行同步，在需要手动恢复时也会保存快照。

a. 您可以按照以下所示配置复制方法"直接"：

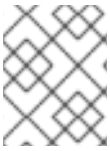
```
$ oc edit cm -n openshift-operators ramen-hub-operator-config
```

b. 将以下内容添加到 **spec.data.ramen_manager_config.yaml** 部分：

```
volsync:
  destinationCopyMethod: Direct
```

4.8. 为测试灾难恢复解决方案创建示例应用程序

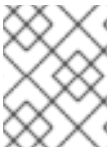
OpenShift Data Foundation 灾难恢复 (DR) 解决方案为由 RHACM 管理的应用程序支持灾难恢复。如需了解更多详细信息，请参阅管理应用程序。



注意

OpenShift Data Foundation DR 解决方案不支持 ApplicationSet，这是通过 ArgoCD 部署的应用程序所需的。

当应用程序在 DRPolicy 中移动应用程序以便进行故障转移或重新定位要求时，ODF DR 使用 PlacementRule 编排 RHACM 应用程序放置。以下小节详细介绍了如何对应用程序应用 **DRPolicy**，以及如果在集群不可用期间或之后管理应用程序放置生命周期。



注意

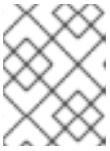
没有 cluster-admin 权限的 OpenShift 用户，请参阅知识库文章 如何为应用程序用户分配必要的权限来执行灾难恢复操作。

4.8.1. 创建示例应用程序

为了测试从 **Primary 受管集群**到 **Secondary 受管集群**的 **failover** 和 **relocate**，我们需要一个简单的应用程序。

先决条件

- 在创建用于常规消耗的应用程序时，请确保应用程序仅部署到一个集群。
- 使用名为 **busybox** 的示例应用作为示例。
- 当应用程序通过或重新定位失败时，确保使用全局流量管理器(GTM)或 Global Server Load Balancing (GLSB)服务配置应用程序的所有外部路由。
- 作为最佳实践，将属于 Red Hat Advanced Cluster Management (RHACM) 订阅的 Red Hat Advanced Cluster Management (RHACM) 订阅分组在一起，以指代单个 Placement Rule 到 DR 保护它们作为组。进一步将它们创建为一个应用程序，用于订阅的逻辑分组，以备将来 DR 操作，如故障转移和重新定位。



注意

如果不相关的订阅引用与放置操作相同的放置规则，它们也会受到 DR 保护，因为 DR 工作流控制引用放置规则的所有订阅。

流程

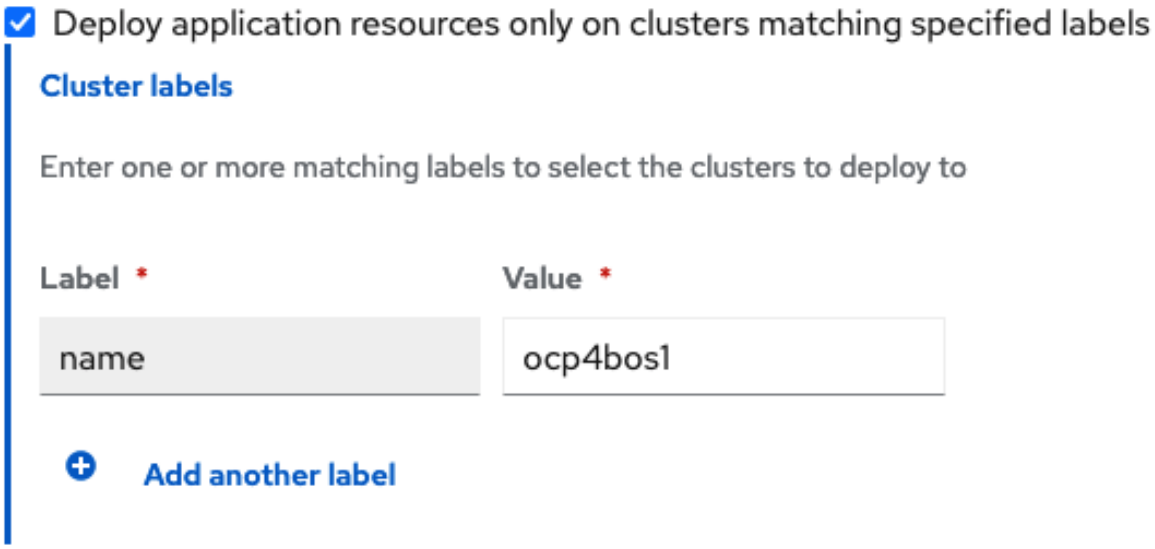
1. 在 Hub 集群中，进入到 **Applications**，再点 **Create application**。
2. 选择类型 **Subscription**。
3. 输入应用程序 **Name** （如 **busybox**）和 **Namespace** （如 **busybox-sample**）。

4. 在 Repository location for resources 部分中，选择 **Repository type Git**。

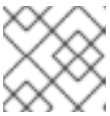
5. 输入示例应用程序的 Git 存储库 URL、github **Branch** 和 **Path**，在其中创建资源 **busybox** Pod 和 PVC。

   - 使用示例应用程序仓库作为 **https://github.com/red-hat-storage/ocm-ramen-samples**
   - 选择 **Branch** 作为 **release-4.12**。
   - 选择以下 **路径**之一
     - **busybox-odr** 使用 RBD Regional-DR。
     - **busybox-odr-cephfs** 使用 CephFS Regional-DR。

6. 滚动表单，直到您看到 **Deploy application resources only on clusters matching specified labels**，然后在 **RHACM** 集群列表视图中添加一个值为 **Primary managed cluster** 名称的标签。



7. 点位于右上角的 **Create**。
   在接下来的屏幕中，进入 **Topology** 选项卡。您可以看到应用程序拓扑上有所有的绿色对勾图标。



注意

要获取更多信息，请点击任何拓扑元素，拓扑视图右侧会显示一个窗口。

8. 验证示例应用程序部署。
   现在，**busybox** 应用程序已部署到首选集群，现在可以验证部署。

   登录到您的受管集群，其中 **busybox** 由 RHACM 部署。

```
$ oc get pods,pvc -n busybox-sample
```

   输出示例：

```
NAME                       READY   STATUS    RESTARTS   AGE
pod/busybox-67bf494b9-zl5tr   1/1     Running   0          77s
```

```
NAME                          STATUS   VOLUME                                     CAPACITY   ACCESS
MODES   STORAGECLASS                AGE
persistentvolumeclaim/busybox-pvc   Bound    pvc-c732e5fe-daaf-4c4d-99dd-462e04c18412
5Gi        RWO            ocs-storagecluster-ceph-rbd   77s
```

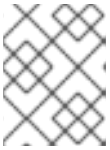
4.8.2. 将 DRPolicy 应用到示例应用程序

先决条件

- 确保 DRPolicy 中引用的两个受管集群都可以访问。如果不能访问，则应用程序在两个集群都在线前无法受到 DR 保护。

流程

1. 在 Hub 集群中，返回到 Multicluster Web 控制台，进入到 **All Clusters**。
2. 登录到 **All Clusters** 下列出的所有集群。
3. 进入到 **Data Services**，然后点 **Data policies**。
4. 点 DRPolicy 末尾的 Actions 菜单，以查看可用操作的列表。
5. 点 **Apply DRPolicy**。
6. 显示 **Apply DRPolicy** 模式时，选择 **busybox** 应用程序，再输入 **PVC** 标签 作为 **appname=busybox**。



注意

如果选择了同一应用程序或多个应用程序下的多个放置规则，则应用程序命名空间中的所有 PVC 都会被默认保护。

7. 点**应用**。
8. 验证在 **Hub 集群**的 **busybox-sample** 命名空间中创建了 **DRPlacementControl** 或 **DRPC**，并它的 **CURRENTSTATE** 显示为 **Deployed**。此资源用于此应用的故障切换和重定位操作。

```
$ oc get drpc -n busybox-sample
```

输出示例：

```
NAME                     AGE     PREFERREDCLUSTER   FAILOVERCLUSTER
DESIREDSTATE   CURRENTSTATE
busybox-placement-1-drpc   6m59s   ocp4bos1                                  Deployed
```

9. [可选] 在主集群中验证 Rados 块设备(RBD) **volumereplication** 和 **volumereplicationgroup**。

```
$ oc get volumereplications.replication.storage.openshift.io
```

输出示例：

```
NAME          AGE     VOLUMEREPLICATIONCLASS              PVCNAME
DESIREDSTATE   CURRENTSTATE
```

```
busybox-pvc      2d16h   rbd-volumereplicationclass-1625360775   busybox-pvc      primary
Primary
```

```
$ oc get volumereplicationgroups.ramendr.openshift.io
```

输出示例：

```
NAME           DESIREDSTATE   CURRENTSTATE
busybox-drpc   primary        Primary
```

10. [可选] 验证 CephFS volsync 复制源已在主集群中成功设置，在故障转移集群中已设置了 VolSync ReplicationDestination。

```
$ oc get replicationsource -n busybox-sample
```

输出示例：

```
NAME           SOURCE         LAST SYNC              DURATION          NEXT SYNC
busybox-pvc    busybox-pvc    2022-12-20T08:46:07Z   1m7.794661104s    2022-12-
20T08:50:00Z
```

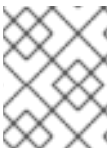
```
$ oc get replicationdestination -n busybox-sample
```

输出示例：

```
NAME           LAST SYNC              DURATION          NEXT SYNC
busybox-pvc    2022-12-20T08:46:32Z   4m39.52261108s
```

4.8.3. 删除示例应用程序

您可以使用 RHACM 控制台删除示例 application **busybox**。



注意

在故障转移和故障恢复（重复）测试完成以及应用程序已可以从 RHACM 和受管集群中删除之前，不应执行示例应用程序的说明。

流程

1. 在 RHACM 控制台上，前往 **Applications**。
2. 搜索要删除的示例应用程序（例如 **busybox**）。
3. 点击您要删除的应用程序旁边的 Action Menu (**⋮**)。
4. 点击 **Delete application**。
   选择了 **Delete application** 时，将显示一个新屏幕，询问是否还应删除与应用相关的资源。
5. 选中 **Remove application 相关资源**复选框，以删除 Subscription 和 PlacementRule。
6. 单击 **Delete**。这将删除主受管集群（或应用程序所运行的任何群集）上的 busybox 应用程序。

7. 除了使用 RHACM 控制台删除资源外，在删除 **busybox** 应用后，还必须立即删除 **DRPlacementControl**。

   a. 登录到 Hub 集群的 OpenShift Web 控制台，再进入为 **busybox-sample** 项目安装的 Operators。

   b. 单击 **OpenShift DR Hub Operator**，然后单击 **DRPlacementControl** 选项卡。

   c. 点击您要删除的 **busybox** 应用程序 DRPlacementControl 旁边的 Action Menu（⋮）。

   d. 单击 **Delete DRPlacementControl**。

   e. 单击 **Delete**。

注意

此过程可用于使用 **DRPlacementControl** 资源删除任何应用。

4.9. 受管集群之间的应用程序故障切换

当受管集群因任何原因而不可用时，执行故障转移。这种故障转移方法是基于应用。

先决条件

- 如果您的设置具有主动和被动 RHACM hub 集群，请参阅使用 Advanced Cluster Management Hub 恢复。
- 当主集群处于 **Ready** 以外的状态时，请检查集群的实际状态，因为它可能需要一些时间才能更新。

  1. 导航到 **RHACM 控制台 → Infrastructure → Clusters → Cluster list**选项卡。

  2. 在执行故障转移操作前，请检查两个受管集群的状态。
     但是，当您故障转移的集群处于 *Ready* 状态时，仍可执行故障转移操作。

流程

1. 在 Hub 集群中，进入到 **Applications**。
2. 点应用程序行末尾的 **Actions** 菜单，以查看可用操作的列表。
3. 点 **Failover application**。
4. 显示 **Failover 应用程序** 弹出窗口中，**选择策略**和**目标集群**，相关的应用程序将在出现灾难时故障转移。
5. 默认情况下，选择通过 复制应用程序资源的订阅组。**点选择订阅组**下拉菜单，验证默认选择或修改此设置。
6. 检查 **Failover readiness** 的状态。

   - 如果状态是 **Ready** 且带有一个绿色勾号，这表示目标集群已就绪，可启动故障转移。继续执行第 7 步。
   - 如果状态是 **Unknown** 或 **Not ready**，请等待到状态变为 **Ready**。

7. 点 **Initiate**。busybox 资源现在在目标集群上创建。
8. 关闭模态窗口，并使用 Applications 页面中的 **Data policies** 列跟踪状态。
9. 验证活动状态是否为应用的 **FailedOver**。
    a. 进入 **Applications → Overview** 选项卡。
    b. 在 **Data policy** 列中，点您要将策略应用到的应用程序的**策略**链接。
    c. 在 **Data policies** 模态页面中，点 **View more details** 链接。
    d. 验证您能否看到一个或多个策略名称以及与应用程序中使用的策略关联的持续活动（同步时间和活动状态）。

4.10. 在受管集群间重新定位应用程序

当所有受管集群都可用时，将应用程序重新定位到首选位置。

前提条件

- 如果您的设置具有主动和被动 RHACM hub 集群，请参阅使用 Advanced Cluster Management Hub 恢复。
- 当主集群处于 **Ready** 以外的状态时，请检查集群的实际状态，因为它可能需要一些时间才能更新。只有在主集群和首选集群启动并运行时才可执行重新定位。
    1. 导航到 **RHACM 控制台 → Infrastructure → Clusters → Cluster list**选项卡。
    2. 在执行重新定位操作前，请单独检查两个受管集群的状态。
- 当最后一次同步时间接近当前时间时，会优先执行重新定位，因为重新定位所需的时间会降低，考虑上次同步时间之间的数据量，现在会有比例更小。
- 在取消隔离前，验证应用程序是否已从集群中清理。

流程

1. 在 Hub 集群中，进入到 **Applications**。
2. 点应用程序行末尾的 **Actions** 菜单，以查看可用操作的列表。
3. 点 **Relocate application**。
4. 当 **Relocate application** 弹出窗口显示时，选择 **policy** 和 **target cluster**，在出现灾难时相关的应用程序将重新定位到其中。
5. 默认情况下，选择部署应用程序资源的订阅组。**点选择订阅组**下拉菜单，验证默认选择或修改此设置。
6. 检查 **Relocation readiness** 的状态。
    - 如果状态是 **Ready** 且带有一个绿色勾号，这表示目标集群已准备好重定位来进行启动。继续执行第 7 步。
    - 如果状态是 **Unknown** 或 **Not ready**，请等待到状态变为 **Ready**。

7. 点 **Initiate**。busybox 资源现在在目标集群上创建。
8. 关闭模态窗口，并使用 Applications 页面中的 **Data policies** 列跟踪状态。
9. 验证应用程序的活动状态是否显示为 **Relocated**。
    a. 进入 **Applications → Overview** 选项卡。
    b. 在 **Data policy** 列中，点您要将策略应用到的应用程序的**策略**链接。
    c. 在 **Data policies** 模态页面中，点 **View more details** 链接。
    d. 验证您能否看到一个或多个策略名称以及与应用程序中使用的策略关联的持续活动（同步时间和活动状态）。

4.11. 查看启用灾难恢复应用程序的恢复点目标值

恢复点目标 (RPO) 值是应用程序当前有效的集群中的持久数据的最新同步时间。此同步时间有助于确定在故障切换过程中丢失的数据持续时间。

注意

此 RPO 值仅适用于故障切换期间的 Regional-DR。重新定位可确保操作过程中没有数据丢失，因为所有对等集群都可用。

您可以在 Hub 集群中查看所有受保护的卷的恢复点目标 (RPO) 值。

流程

1. 在 Hub 集群中，进入到 **Applications → Overview** 选项卡。
2. 在 **Data policy** 列中，点您要将策略应用到的应用程序的**策略**链接。
   Data policies modal 页面会出现，每个应用程序所应用的灾难恢复策略数量以及故障转移和重新定位状态。
3. 在 **Data policies** 模态页面中，点 **View more details** 链接。
   此时会显示一个详细的 **数据策略** 模态页面，其中显示策略名称和与应用到应用程序的策略关联的持续活动(Last sync、Activity 状态)。

   在模态页面中报告的 **Last sync 时间** 代表，表示所有针对应用程序保护的 DR 的卷的最新同步时间。

第 5 章 使用 ADVANCED CLUSTER MANAGEMENT HUB 恢复

5.1. 故障转移注意事项

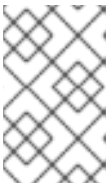
当您的设置有主动和被动 RHACM hub 集群时：

1. 在恢复前，在被动 RHACM hub 上安装多集群集群编配器(MCO) Operator。有关恢复 RHACM hub 的说明，请参阅准备新的 hub 集群。
2. 确保 **Restore.cluster.open-cluster-management.io** 资源将 **.spec.cleanupBeforeRestore** 设置为 **None**。
3. 如果在设置前手动配置了跨集群的 SSL 访问，则在集群中重新配置 SSL 访问。具体步骤请参阅解决方案中的 *在集群中配置 SSL 访问* 部分。
4. 安装 MCO Operator 后，要恢复备份，请参阅从备份中恢复 hub 集群。
5. 恢复完成后，等待 DRPolicy 验证成功。
6. 验证 **DRPolicy** 是否已成功创建。在 **Hub 集群**中，为创建的每个 DRPolicy 资源运行这个命令，使用您的唯一名称替换 *<drpolicy_name>*。

   ```
   oc get drpolicy <drpolicy_name> -o jsonpath='{.status.conditions[].reason}{"\n"}'
   ```

   输出示例：

   ```
   Succeeded
   ```

   注意

   将失败的 hub 恢复到其被动实例将仅在最后一次调度的备份中恢复应用程序及其 DR 保护状态。在最后一次调度的备份后，任何受 DR 保护的应用程序都需要在新 hub 上再次保护。

7. 对于后续步骤，请转至解决方案的故障切换部分。

5.2. 重新定位注意事项

当您的设置有主动和被动 RHACM hub 集群时：

1. 如果您在新 hub 中恢复了数据，请等待 RHACM 控制台中两个受管集群都处于 **Ready** 状态。如需更多信息，请参阅恢复导入的受管集群。
2. 如果在设置前手动配置了跨集群的 SSL 访问，则在集群中重新配置 SSL 访问。具体步骤请参阅解决方案中的 *在集群中配置 SSL 访问* 部分。
3. 在重新定位到首选集群前，
   a. 检查 PROGRESSION 状态。如果它一直处于 **Cleaning Up（清理）**状态，则手动清理首选集群中的应用程序。

      ```
      $ oc get drpc  -n busybox -owide
      ```

```
NAME                        AGE   PREFERREDCLUSTER   FAILOVERCLUSTER
DESIREDSTATE   CURRENTSTATE   PROGRESSION   START TIME
DURATION   PEER READY
busybox-placement-1-drpc        10h   ocp4perf1          ocp4perf2          Failover
FailedOver     Cleaning Up   2023-03-07T05:10:39Z             False
```

b. 删除首选集群中的孤立的 **AppliedManifestWork**。您可以通过查看前导 **sha256** 值来查找孤立的 **AppliedManifestWork**。

```
$ echo -n "https://api.ocp4perf-hub.qe.rh-ocs.com:6443" | sha256sum
```

输出示例：

```
7044ffd168d05a6d28d2f6ac5a99ad8693c
```

在这种情况下，旧 hub 的服务器 url 是 https://api.ocp4perf-hub.qe.rh-ocs.com:6443，必须删除的 **AppliedManfiestWork** 是 **7044ffd168d05a6d28d2f6ac5a99ad8693c-busybox-sample-subscription**

```
$ oc delete AppliedManifestWork 7044ffd168d05a6d28d2f6ac5a99ad8693c-busybox-
sample-subscription
```

c. 在继续重新定位前，验证您的应用程序是否已在首选集群中删除。

注意：将失败的 hub 恢复到其被动实例，将仅在最后一次调度的备份中恢复应用程序及其 DR 保护状态。在最后一次调度的备份后，任何受 DR 保护的应用程序都需要在新 hub 上再次保护。

4. 对于后续步骤，请转至解决方案的重定位部分。

第 6 章 灾难恢复故障排除

6.1. 对 METRO-DR 进行故障排除

6.1.1. 在故障切换后，有状态集应用程序会卡住

问题

在重新定位到首选集群时，DRPlacementControl 会一直报告 PROGRESSION 为 "MovingToSecondary"。
在以前的版本中，在 Kubernetes v1.23 之前，Kubernetes control plane 不会清理为 StatefulSets 创建的 PVC。此活动由集群管理员或管理 StatefulSets 的软件操作员保留。因此，当 Pod 被删除时，StatefulSet 的 PVC 会保持不变。这可防止 Ramen 将应用程序重新定位到首选集群。

解决方案

1. 如果工作负载使用 StatefulSets，且重新定位会卡住 PROGRESSION 为 "MovingToSecondary"，则运行：

```
$ oc get pvc -n <namespace>
```

2. 对于属于该 StatefulSet 的命名空间的每个绑定的 PVC，请运行

```
$ oc delete pvc <pvcname> -n namespace
```

删除所有 PVC 后，卷组复制组 (VRG) 过渡到 secondary，然后会被删除。

3. 运行以下命令

```
$ oc get drpc -n <namespace> -o wide
```

几分钟后，PROGRESSION 报告 "Completed" 并完成重新定位。

结果

工作负载重新定位到首选集群

BZ reference: [2118270]

6.1.2. DR 策略可保护同一命名空间中的所有应用程序

问题

虽然只有单个应用程序被选择供 DR 策略使用，但同一命名空间中的所有应用程序都会受到保护。这会导致 PVC，与多个工作负载的 **DRPlacementControl spec.pvcSelector** 匹配；或者，如果所有工作负载中没有选择器，复制管理可能会多次管理每个 PVC，并根据单独的 **DRPlacementControl** 操作造成数据崩溃或无效操作。

解决方案

标签 PVC 属于唯一工作负载，并使用所选标签作为 DRPlacementControl **spec.pvcSelector**，以忽略哪个 DRPlacementControl 保护和管理命名空间中的哪些 PVC 子集。无法使用用户界面为 DRPlacementControl 指定 **spec.pvcSelector** 字段，因此必须使用命令行删除并创建此类应用程序的 DRPlacementControl。

BZ 参考： [2111163]

6.1.3. 在应用程序故障时处于 Relocating 状态

问题

这个问题可能会在执行故障转移和应用程序的故障后发生（所有节点或集群）。当执行故障恢复应用程序时，会停留在 **Relocating** 状态，并显示 **Waiting** for PV restore to complete。

解决方案

使用 S3 客户端或等同从 s3 存储清理重复的 PV 对象。仅保持接近故障转移或重新定位时间的时间戳的那一个。

BZ 参考： [2120201]

6.1.4. 无法使用 RHACM 2.8 将 DRPolicy 应用到订阅工作负载

问题

Red Hat Advanced Cluster Management (RHACM) 2.8 控制台已弃用 **PlacementRule** 类型，并移到 Subscription 应用程序的 **放置类型**。因此，当用户使用 RHACM 2.8 控制台创建 Subscription 应用程序时，只会使用 Placement 创建应用程序。由于 OpenShift Data Foundation 4.12 灾难恢复用户界面和 Ramen 操作器不支持订阅应用程序的放置，所以灾难恢复用户界面无法检测应用程序并显示分配策略的详情。

解决方案

由于 RHACM 2.8 控制台仍能够检测 **PlacementRule** （使用命令行界面(CLI)创建的 PlacementRule），请执行以下步骤，以使用 **PlacementRule** 在 RHACM 2.8 中创建 Subscription 应用程序：

1. 使用应用程序命名空间创建一个新项目。（例如： **busybox-application**）
2. 找到要部署应用的受管集群的标签（例如： **drcluster1-jul-6**）
3. 使用上一步中创建的受管集群标签在 **application-namespace** 上创建 **PlacementRule** CR：

```
apiVersion: apps.open-cluster-management.io/v1
kind: PlacementRule
metadata:
  labels:
    app: busybox-application
  name: busybox-application-placementrule-1
  namespace: busybox-application
spec:
  clusterSelector:
    matchLabels:
      name: drcluster1-jul-6
```

4. 在使用 Subscription 应用程序页面中的 RHACM 控制台创建应用程序时，请选择此新的 **PlacementRule**。
5. 从 YAML 编辑器删除 **PlacementRule**，以便可以重新使用所选编辑器。

BZ 参考： [2216190]

6.2. 对 REGIONAL-DR 进行故障排除

6.2.1. 为某些镜像停止 RBD 镜像调度

问题

对于某些镜像，需要一些常见原因来进行 RBD 镜像调度停止。
在为镜像标记应用程序后，出于某种原因（如果不复制），请使用 toolbox pod 并运行它，以查看哪些镜像调度已停止。

```
$ rbd snap ls <poolname/imagename> –all
```

解决方案

- 在主集群中重启 manager 守护进程
- 在主集群中禁用并立即重新启用受影响镜像的 mirroring

BZ 参考：[2067095 和 2121514]

6.2.2. rbd-mirror 守护进程健康状态

问题

当 mirror 服务 **::get_mirror_service_status** 调用 **Ceph** monitor 来获取 **rbd-mirror** 的服务状态，会有多种情况会报告 WARNING。
在网络连接网络连接后，**rbd-mirror** 守护进程健康状态在 **警告** 状态，同时两个受管集群间的连接都正常。

解决方案

在 toolbox 中运行以下命令并查找 **leader:false**

```
rbd mirror pool status --verbose ocs-storagecluster-cephblockpool | grep 'leader:'
```

如果您在输出中看到以下内容：

leader: false	这表示存在守护进程启动问题，最有可能的根本原因是由于问题可靠地连接到 secondary 集群。 临时解决方案：通过删除 pod 并验证它已重新调度到另一节点上，将 **rbd-mirror** pod 移到另一个节点。
leader: true 或没有输出	联系红帽支持。

BZ 参考：[2118627]

6.2.3. 在故障切换后，有状态集应用程序会卡住

问题

在重新定位到首选集群时，DRPlacementControl 会一直报告 PROGRESSION 为 "MovingToSecondary"。

在以前的版本中，在 Kubernetes v1.23 之前，Kubernetes control plane 不会清理为 StatefulSets 创建的 PVC。此活动由集群管理员或管理 StatefulSets 的软件操作员保留。因此，当 Pod 被删除时，StatefulSet 的 PVC 会保持不变。这可防止 Ramen 将应用程序重新定位到首选集群。

解决方案

1. 如果工作负载使用 StatefulSets，且重新定位会卡住 PROGRESSION 为 "MovingToSecondary"，则运行：

```
$ oc get pvc -n <namespace>
```

2. 对于属于该 StatefulSet 的命名空间的每个绑定的 PVC，请运行

```
$ oc delete pvc <pvcname> -n namespace
```

删除所有 PVC 后，卷组复制组 (VRG) 过渡到 secondary，然后会被删除。

3. 运行以下命令

```
$ oc get drpc -n <namespace> -o wide
```

几分钟后，PROGRESSION 报告 "Completed" 并完成重新定位。

结果

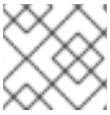
工作负载重新定位到首选集群

BZ reference: [2118270]

6.2.4. 故障转移后应用程序没有运行

问题

在应用程序失败时，工作负载 pod 无法访问运行状态，并带有错误 **MountVolume.MountDevice failed for volume <PV name> : rpc error: code = Internal desc = fail to check rbd image status: *(cannot map image <image description> it is not primary)***



注意

在工作负载失败的集群中执行这些步骤。

解决方案

1. 将 RBD 镜像守护进程部署缩减为 **0**，直到应用 pod 可以从上述错误中恢复。

```
$ oc scale deployment rook-ceph-rbd-mirror-a -n openshift-storage --replicas=0
```

2. 在恢复后，将 RBD 镜像守护进程部署重新调度到 **1**。

```
$ oc scale deployment rook-ceph-rbd-mirror-a -n openshift-storage --replicas=1
```

BZ 参考：[2134936]

6.2.5. volsync-rsync-src pod 处于错误状态

问题

volsync-rsync-src pod 处于错误状态，因为它们无法连接到 **volsync-rsync-dst**。**VolSync** 源 pod 日志可能会在与日志片断类似的延长持续时间内显示持久性错误消息。
运行以下命令来检查日志。

```
$ oc logs volsync-rsync-src-<app pvc name>-<suffix>
```

输出示例

```
VolSync rsync container version: ACM-0.6.0-ce9a280
Syncing data to volsync-rsync-dst-busybox-pvc-9.busybox-workloads-1.svc.clusterset.local:22
Syncronization failed. Retrying in 2 seconds. Retry 1/5.
rsync: connection unexpectedly closed (7 bytes received so far) [sender]
rsync error: unexplained error (code 255) at io.c(226) [sender=3.1.3]
```

解决方案

您可以按照以下步骤重新配置最大传输单元 (MTU) 大小来解决这个问题：

1. 注解具有 submariner 网关标签的节点。

   ```
   $ oc annotate node -l submariner.io/gateway submariner.io/tcp-clamp-mss=1340 --
   overwrite
   ```

   输出示例

   ```
   node/compute-0 annotated
   node/compute-2 annotated
   ```

2. 删除 submariner 路由代理 pod。

   ```
   $ oc delete pods -n submariner-operator -l app=submariner-routeagent
   ```

   输出示例

   ```
   pod "submariner-routeagent-4r66z" deleted
   pod "submariner-routeagent-4tn6d" deleted
   pod "submariner-routeagent-9r42l" deleted
   pod "submariner-routeagent-bg5wq" deleted
   pod "submariner-routeagent-gzqdj" deleted
   pod "submariner-routeagent-j77jq" deleted
   ```

3. 检查 **vol-sync-src pod** 中是否存在任何错误。

   ```
   $ oc logs volsync-rsync-src-dd-io-pvc-3-nwn8h
   ```

   输出示例

   ```
   VolSync rsync container version: ACM-0.6.0-ce9a280
   Syncing data to volsync-rsync-dst-dd-io-pvc-3.busybox-workloads-
   8.svc.clusterset.local:22 …
   ```

```
.d..tp..... ./
<f+++++++++ 07-12-2022_13-03-04-dd-io-3-5d6b4b84df-v9bhc
```

BZ 参考： [2136864]

6.2.6. volsync-rsync-src pod 处于错误状态，因为它无法解析目标主机名

问题

VolSync 源 pod 无法解析 VolSync 目标 pod 的主机名。VolSync Pod 的日志在延长时间内显示错误消息，类似于以下日志片断。

```
$ oc logs -n busybox-workloads-3-2 volsync-rsync-src-dd-io-pvc-1-p25rz
```

输出示例

```
VolSync rsync container version: ACM-0.6.0-ce9a280
Syncing data to volsync-rsync-dst-dd-io-pvc-1.busybox-workloads-3-2.svc.clusterset.local:22 ...
ssh: Could not resolve hostname volsync-rsync-dst-dd-io-pvc-1.busybox-workloads-3-
2.svc.clusterset.local: Name or service not known
```

解决方案

在这两个节点上，重启 **submariner-lighthouse-agent**。

```
$ oc delete pod -l app=submariner-lighthouse-agent -n submariner-operator
```

6.2.7. 无法使用 RHACM 2.8 将 DRPolicy 应用到订阅工作负载

问题

Red Hat Advanced Cluster Management (RHACM) 2.8 控制台已弃用 **PlacementRule** 类型，并移到 Subscription 应用程序的 **放置类型**。因此，当用户使用 RHACM 2.8 控制台创建 Subscription 应用程序时，只会使用 Placement 创建应用程序。由于 OpenShift Data Foundation 4.12 灾难恢复用户界面和 Ramen 操作器不支持订阅应用程序的放置，所以灾难恢复用户界面无法检测应用程序并显示分配策略的详情。

解决方案

由于 RHACM 2.8 控制台仍能够检测 **PlacementRule** （使用命令行界面(CLI)创建的 PlacementRule），请执行以下步骤，以使用 **PlacementRule** 在 RHACM 2.8 中创建 Subscription 应用程序：

1. 使用应用程序命名空间创建一个新项目。（例如： **busybox-application**）
2. 找到要部署应用的受管集群的标签（例如： **drcluster1-jul-6**）
3. 使用上一步中创建的受管集群标签在 **application-namespace** 上创建 **PlacementRule** CR：

```
apiVersion: apps.open-cluster-management.io/v1
kind: PlacementRule
metadata:
  labels:
    app: busybox-application
  name: busybox-application-placementrule-1
```

```
  namespace: busybox-application
spec:
  clusterSelector:
    matchLabels:
      name: drcluster1-jul-6
```

4. 在使用 Subscription 应用程序页面中的 RHACM 控制台创建应用程序时，请选择此新的 **PlacementRule**。

5. 从 YAML 编辑器删除 **PlacementRule**，以便可以重新使用所选编辑器。

BZ 参考：[2216190]