Red Hat OpenShift Data Foundation 4.10

为带有 Advanced Cluster Management 的 Regional-DR 配置 OpenShift Data Foundation

开发者预览:有关使用 Regional-DR 功能设置 OpenShift Data Foundation 的说明。此解决方案是一个开发者技术预览功能，它不应该在生产环境中运行。

Last Updated: 2023-09-19

Red Hat OpenShift Data Foundation 4.10 为带有 Advanced Cluster Management 的 Regional-DR 配置 OpenShift Data Foundation

开发者预览:有关使用 Regional-DR 功能设置 OpenShift Data Foundation 的说明。此解决方案是一个开发者技术预览功能，它不应该在生产环境中运行。

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。 Configuring OpenShift Data Foundation for Regional-DR with Advanced Cluster Management is a Developer Preview feature and is subject to Developer Preview support limitations. Developer Preview releases are not intended to be run in production environments and are not supported through the Red Hat Customer Portal case management system. If you need assistance with Developer Preview features, reach out to the ocs-devpreview@redhat.com mailing list and a member of the Red Hat Development Team will assist you as quickly as possible based on their availability and work schedules.

目录

让开源更具包容性 3

对红帽文档提供反馈 4

第 1 章 REGIONAL-DR 介绍 5
1.1. 区域 DR 解决方案的组件 5
1.2. 区域DR 部署工作流 6

第 2 章 启用区域DR 的要求 8

第 3 章 在受管集群中安装 OPENSHIFT DATA FOUNDATION 10

第 4 章 在 HUB 集群上安装 OPENSHIFT DR HUB OPERATOR 11

第 5 章 配置多站点存储复制 12
5.1. 安装 OPENSHIFT DATA FOUNDATION 多集群编排器 12
5.2. 在 HUB 集群中创建镜像对等 12
5.3. 在受管集群中验证 CEPH 镜像 13
5.4. 验证对象存储桶和 S3STOREPROFILES 14

第 6 章 创建镜像 STORAGECLASS 资源 16

第 7 章 配置 S3 端点之间的 SSL 访问 17

第 8 章 在 HUB 集群上创建灾难恢复策略 19

第 9 章 启用自动安装 OPENSHIFT DR 集群 OPERATOR 21

第 10 章 启用自动向受管集群传输 S3SECRETS 22

第 11 章 创建示例应用程序 23
11.1. 删除示例应用程序 26

第 12 章 受管集群之间的应用程序故障切换 28

第 13 章 在受管集群间重新定位应用程序 31

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。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**。

第 1 章 REGIONAL-DR 介绍

灾难恢复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连续性战略，旨在在重大危险事件期间保持业务运营的连续性。

Regional-DR（区域 DR） 功能在地理分散的网站之间提供卷持久数据和元数据复制。在公共云中，它们类似于防止区域故障。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）。这通常通过 Recovery Point Objective (RPO) 和 Recovery Time Objective (RTO) 代表。

- 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配置基础架构以启用灾难恢复所需的步骤和命令。

1.1. 区域 DR 解决方案的组件

region-DR 由 Red Hat Advanced Cluster Management for Kubernetes(RHACM)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 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

有关该产品的更多信息，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 堆栈有了以下改进：

- 启用用于镜像的池
- 在 RBD 块池中自动镜像镜像
- 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 集群的应用状态关系
- 在应用程序状态变为对等集群时失败
- 将应用的状态重新定位到之前部署的集群

OpenShift DR 被分成三个组件：

- **ODF 多集群编排器** ：在多集群 control plane(RHACM Hub)上安装，它还执行以下操作：
  - 创建 bootstrap 令牌并在受管集群间交换此令牌。
  - 在受管集群中为默认的 **CephBlockPool** 启用镜像。
  - 为每个受管集群上的 **PVC** 和 **PV** 元数据创建一个使用 Multicloud Object Gateway(MCG)的对象存储桶。
  - 为每个新对象存储桶创建一个 **Secret**，其中包含 **openshift-dr-system** 项目中对 **Hub 集群**的存储桶访问的密钥。
  - 在**主受管集群**和**次受管集群**上创建 **VolumeReplicationClass**，间隔为 **schedulingIntervals**（例如 5m, 15m, 30m）。
  - 修改 Hub 集群上的 **ramen-hub-operator-config** ConfigMap，并添加 s3StoreProfiles 条目。
- **OpenShift DR Hub Operator**:安装在 hub 集群上，为应用程序管理故障转移和重定位。
- **OpenShift DR Cluster Operator**:安装在每个受管集群上，以管理应用程序的所有 PVC 的生命周期。

1.2. 区域DR 部署工作流

本节概述在两个不同的 OpenShift Container Platform 集群中使用 OpenShift Data Foundation 版本 4.10 和 RHACM 的最新版本来配置和部署 Regional-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 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. 确保您满足每个 Regional-DR 要求，包括 RHACM 操作器安装、创建或导入 OpenShift Container Platform 到 RHACM hub 和网络配置。请参阅启用区域DR 的要求。
2. 在 Primary 和 Secondary 受管集群中安装 OpenShift Data Foundation 4.10。请参阅在受管集群中安装 OpenShift Data Foundation。
3. 在 Hub 集群上安装 Openshift DR Hub Operator。请参阅在 Hub 集群上安装 OpenShift DR Hub Operator。
4. 通过在两个 OpenShift Data Foundation 受管集群之间创建镜像关系来配置多站点存储复制。请参阅配置多站点存储复制。
5. 在每个受管集群上创建镜像 StorageClass 资源，支持新的 **imageFeatures** 以实现启用镜像(mirror)的块卷。请参阅创建镜像 StorageClass 资源。
6. 在 hub 集群上创建 DRPolicy 资源，用于在受管集群间部署、故障转移和重新定位工作负载。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。

7. 启用 OpenShift DR Cluster Operator 自动安装，并在受管集群中自动传输 S3 secret。具体步骤请参阅 启用 OpenShift DR 集群 Operator 的自动安装，并 在受管集群中启用 S3 secret 自动传输。

8. 使用 RHACM 控制台创建示例应用程序，用于测试故障转移和重定位测试。具体步骤，请参阅 创建示例应用程序、应用程序故障切换并在受管集群间重新定位应用程序。

第 2 章 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：

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

  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。
- 您必须有三个在它们之间具有网络可访问性的 OpenShift 集群：
  - 安装 Advanced Cluster Management for Kubernetes(RHACM operator)、ODF Multicluster Orchestrator 和 OpenShift DR Hub 控制器的 **hub 集群**。
  - 安装 **OpenShift Data Foundation、OpenShift DR 集群控制器和应用的主要受管集群**。
  - 安装 **OpenShift Data Foundation、OpenShift DR 集群控制器和应用的辅助受管集群**。
- 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。
  - 使用您的 OpenShift 凭证登录到 RHACM 控制台。
  - 查找为 Advanced Cluster Manager 控制台创建的路由：

    ```
    $ oc get route multicloud-console -n open-cluster-management -o jsonpath --
    template="https://{.spec.host}/multicloud/clusters{'\n'}"
    ```

    输出示例：

    ```
    https://multicloud-console.apps.perf3.example.com/multicloud/clusters
    ```

    使用 OpenShift 凭据登录后，您应该会看到导入的本地集群。
- 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**。
- 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
  ```

  cluster1 的输出示例（如 **ocp4perf1**）：

  ```
  {
    "clusterNetwork": [
      {
        "cidr": "10.5.0.0/16",
        "hostPrefix": 23
      }
    ],
  ```

```
  "externalIP": {
    "policy": {}
  },
  "networkType": "OpenShiftSDN",
  "serviceNetwork": [
    "10.15.0.0/16"
  ]
}
```

cluster2 的输出示例（如 **ocp4perf2**）：

```
{
  "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。

第 3 章 在受管集群中安装 OPENSHIFT DATA FOUNDATION

流程

1. 在每个受管集群上安装 OpenShift Data Foundation 版本 4.10。
   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上是否成功部署：

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

   对于 Multicloud 对象网关(MCG)：

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

   如果状态结果在**主受管集群**和**二级受管集群**上都处于 **Ready** 状态，则继续在受管集群上启用镜像(mirror)。

第 4 章 在 HUB 集群上安装 OPENSHIFT DR HUB OPERATOR

步骤

1. 在 Hub 集群中，导航到 OperatorHub 并使用 **OpenShift DR Hub Operator** 的搜索过滤器。
2. 按照屏幕说明，将操作器安装到 **openshift-dr-system** 项目中。
3. 使用以下命令验证 Operator Pod 是否处于 **Running** 状态：

```
$ oc get pods -n openshift-dr-system
```

输出示例：

```
NAME                                  READY   STATUS    RESTARTS   AGE
ramen-hub-operator-898c5989b-96k65    2/2     Running   0          4m14s
```

第 5 章 配置多站点存储复制

镜像或复制以每个 **CephBlockPool** 为基础在对等受管集群中启用，然后可根据池中的特定镜像子集进行配置。**rbd-mirror** 守护进程负责将本地对等集群的镜像更新复制到远程集群中的同一镜像。

这些说明详细介绍了如何在两个 OpenShift Data Foundation 受管集群之间创建镜像关系。

5.1. 安装 OPENSHIFT DATA FOUNDATION 多集群编排器

OpenShift Data Foundation Multicluster Orchestrator 是一个控制器，从 Hub 集群上的 OpenShift Container Platform OperatorHub 安装。这个多集群编排器控制器以及 MirrorPeer 自定义资源会创建一个 bootstrap 令牌，并在受管集群间交换这个令牌。

流程

1. 导航到 **Hub 集群**上的 **OperatorHub**，并使用关键字过滤器搜索 **ODF Multicluster Orchestrator**。
2. 点击 **ODF Multicluster Orchestrator** 标题。
3. 保留所有默认设置并单击 Install。
   operator 资源安装在 **openshift-operators** 中，并可用于所有命名空间。
4. 验证 **ODF Multicluster Orchestrator** 是否已成功安装。
   a. 通过能够选择 View Operator 来验证安装是否成功。
   b. 验证 Operator Pod 是否处于 **Running** 状态。

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

输出示例：

```
NAME                                  READY   STATUS    RESTARTS   AGE
odfmo-controller-manager-65946fb99b-779v8   1/1     Running   0          5m3s
```

5.2. 在 HUB 集群中创建镜像对等

mirror Peer 是一个集群范围的资源，用于保存有关将具有对等对等关系的受管集群的信息。

先决条件

- 确保 **Hub 集群**上安装了 **ODF 多集群编排器**。
- 每个镜像对等点只能有两个集群。
- 确保每个集群都有唯一标识的集群名称，如 **ocp4perf1** 和 **ocp4perf2**。

流程

1. 点击 **ODF Multicluster Orchestrator** 查看 Operator 详情。
   您还可以在成功安装 Multicluster Orchestrator 后点 **View Operator**。
2. 点击 Mirror Peer API **Create 实例**，然后选择 **YAML** 视图。

3. 在将 *<cluster1>* 和 *<cluster2>* 替换为 RHACM 控制台中受管集群的正确名称后，将以下 YAML 复制到一个名为 **mirror-peer.yaml** 的文件。

```
apiVersion: multicluster.odf.openshift.io/v1alpha1
kind: MirrorPeer
metadata:
  name: mirrorpeer-<cluster1>-<cluster2>
spec:
  items:
  - clusterName: <cluster1>
    storageClusterRef:
      name: ocs-storagecluster
      namespace: openshift-storage
  - clusterName: <cluster2>
    storageClusterRef:
      name: ocs-storagecluster
      namespace: openshift-storage
  manageS3: true
  schedulingIntervals:
  - 5m
  - 15m
```

注意

schedulingIntervals 的时间值（如 5m）将用于配置复制持久性卷所需的间隔。对于关键应用程序，这些值可映射到您的恢复点目标(RPO)。修改 **schedulingIntervals** 中的值，以适合您的应用程序要求。最小值为 **1m**，默认值为 **5m**。

4. 将唯一 **mirror-peer.yaml** 文件的内容复制到 **YAML** 视图中。您必须完全替换原始的内容。

5. 点 YAML 视图屏幕底部的 **Create**。

6. 在继续操作前，验证您是否可以将 **Phase** 状态显示为 **ExchangedSecret**。

5.3. 在受管集群中验证 CEPH 镜像

在**主受管集群**和**二级受管集群**上执行以下验证，以检查 Ceph 镜像是否活跃：

1. 验证 **mirroring** 在默认 **Ceph 块池**中被启用。

```
$ oc get cephblockpool -n openshift-storage -o=jsonpath='{.items[?
(@.metadata.ownerReferences[*].kind=="StorageCluster")].spec.mirroring.enabled}{"\n"}'
```

输出示例：

```
true
```

2. 验证 **rbd-mirror** Pod 正在运行。

```
$ oc get pods -o name -l app=rook-ceph-rbd-mirror -n openshift-storage
```

输出示例：

```
pod/rook-ceph-rbd-mirror-a-6486c7d875-56v2v
```

3. 检查 **守护进程** 健康状态，以确保它正常。

```
$ 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":{}}
```

注意

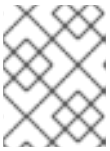
守护进程 健康和健康字段可能需要 10 分钟时间才会从 **Warning** 更改为 **OK**。如果 10 分钟后状态没有变为 OK，则使用 Advanced Cluster Manager 控制台验证 **submariner 附加组件**连接是否仍处于健康状态。

4. 验证 **VolumeReplicationClass** 是否在主 **受管集群**和**二级受管集群**上为每个在 MirrorPeer 中列出的每个 schedulingIntervals 创建集群（如 5m，15m）。

```
$ oc get volumereplicationclass
```

输出示例：

```
NAME                              PROVISIONER
rbd-volumereplicationclass-1625360775   openshift-storage.rbd.csi.ceph.com
rbd-volumereplicationclass-539797778    openshift-storage.rbd.csi.ceph.com
```



注意

VolumeReplicationClass 用于指定要复制的每个卷的 **mirroringMode**，以及将卷或镜像从本地集群复制到远程集群的频率（例如，每 5 分钟一次）。

5.4. 验证对象存储桶和 S3STOREPROFILES

在**主受管集群**和**二级受管集群**上执行以下验证，以检查 Ceph 镜像是否活跃。

流程

1. 验证在 **openshift-storage** 命名空间中的**主受管集群**和**二级受管集群**中有新的 **Object Bucket Claim** 以及相应的 **Object Bucket**。

```
$ oc get obc,ob -n openshift-storage
```

输出示例：

```
NAME                                                    STORAGE-CLASS                 PHASE   AGE
objectbucketclaim.objectbucket.io/odrbucket-21eb5332f6b6   openshift-storage.noobaa.io
Bound   13m

NAME                                                          STORAGE-CLASS                 CLAIM-
```

```
NAMESPACE   CLAIM-NAME   RECLAIM-POLICY   PHASE   AGE
objectbucket.objectbucket.io/obc-openshift-storage-odrbucket-21eb5332f6b6   openshift-
storage.noobaa.io                          Delete          Bound   13m
```

2. 验证 **Hub cluster openshift-dr-system** 命名空间中有两个新的 **Secret**，其中包含到每个新 Object Bucket Class 的 access 和 secret key。

```
$ oc get secrets -n openshift-dr-system | grep Opaque
```

输出示例：

```
8b3fb9ed90f66808d988c7edfa76eba35647092   Opaque  2      16m
af5f82f21f8f77faf3de2553e223b535002e480   Opaque  2      16m
```

3. OBC 和 Secret 在被写入到新创建的 **s3StoreProfiles** 部分的 Hub 集群上的 ConfigMap **ramen-hub-operator-config** 中。

```
$ oc get cm ramen-hub-operator-config -n openshift-dr-system -o yaml | grep -A 14
s3StoreProfiles
```

输出示例：

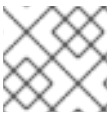
```
s3StoreProfiles:
- s3Bucket: odrbucket-21eb5332f6b6
  s3CompatibleEndpoint: https://s3-openshift-storage.apps.perf2.example.com
  s3ProfileName: s3profile-ocp4perf2-ocs-storagecluster
  s3Region: noobaa
  s3SecretRef:
    name: 8b3fb9ed90f66808d988c7edfa76eba35647092
    namespace: openshift-dr-system
- s3Bucket: odrbucket-21eb5332f6b6
  s3CompatibleEndpoint: https://s3-openshift-storage.apps.perf1.example.com
  s3ProfileName: s3profile-ocp4perf1-ocs-storagecluster
  s3Region: noobaa
  s3SecretRef:
    name: af5f82f21f8f77faf3de2553e223b535002e480
    namespace: openshift-dr-system
```

注意

记录下 **s3ProfileName** 的名称。它们将在 DRPolicy 资源中使用。

第 6 章 创建镜像 STORAGECLASS 资源

必须使用新的 StorageClass 创建启用了**镜像**的块卷，该卷有额外的 **imageFeatures** 在受管集群之间启用更快的镜像复制。新功能包括 *exclusive-lock*、*object-map* 和 *fast-diff*。默认 OpenShift Data Foundation **StorageClass ocs-storagecluster-ceph-rbd** 不包括这些功能。



注意

此资源必须在**主受管集群**和**次受管集群**上创建。

流程

1. 将以下 YAML 保存到文件 **ocs-storagecluster-ceph-rbdmirror.yaml**。

```
allowVolumeExpansion: true
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ocs-storagecluster-ceph-rbdmirror
parameters:
  clusterID: openshift-storage
  csi.storage.k8s.io/controller-expand-secret-name: rook-csi-rbd-provisioner
  csi.storage.k8s.io/controller-expand-secret-namespace: openshift-storage
  csi.storage.k8s.io/fstype: ext4
  csi.storage.k8s.io/node-stage-secret-name: rook-csi-rbd-node
  csi.storage.k8s.io/node-stage-secret-namespace: openshift-storage
  csi.storage.k8s.io/provisioner-secret-name: rook-csi-rbd-provisioner
  csi.storage.k8s.io/provisioner-secret-namespace: openshift-storage
  imageFeatures: layering,exclusive-lock,object-map,fast-diff
  imageFormat: "2"
  pool: ocs-storagecluster-cephblockpool
provisioner: openshift-storage.rbd.csi.ceph.com
reclaimPolicy: Delete
volumeBindingMode: Immediate
```

2. 在两个受管集群中创建文件。

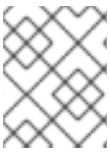
```
$ oc create -f ocs-storagecluster-ceph-rbdmirror.yaml
```

输出示例：

```
storageclass.storage.k8s.io/ocs-storagecluster-ceph-rbdmirror created
```

第 7 章 配置 S3 端点之间的 SSL 访问

配置 **s3 端点**之间的网络(SSL)访问，以便元数据可以在 **MCG 对象存储桶**中存储，使用安全传输协议并在 **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. 在**主受管集群**, **二级受管集群**, 和 **Hub cluster** 上创建一个新的 **ConfigMap** 用于保存远程集群的证书捆绑，其文件名为 **cm-clusters-crt.yaml**。

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**主受管集群**、**二级受管集群**和 **Hub 集群**上创建 ConfigMap 文件。

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

输出示例：

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

重要

对于 Hub 集群，使用 **DRPolicy** 资源验证对象存储桶的访问权限，必须在 Hub 集群上创建相同的 **ConfigMap cm-clusters-crt.yaml**。

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
```

第 8 章 在 HUB 集群上创建灾难恢复策略

OpenShift DR 使用 RHACM hub 集群上的 Disaster Recovery Policy(DRPolicy)资源（集群范围）来部署、故障转移和重新定位受管集群中的工作负载。

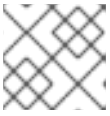
先决条件

- 确保存在两个集群，它们对等进行存储级别复制，并且启用了 CSI 卷复制。
- 确保有一个调度间隔，决定按照什么频率执行数据复制，它也充当使用 DRPolicy 的工作负载的粗粒度恢复点目标(RPO)。
- 确保为策略中的每个集群分配 S3 配置文件名称，该名称使用 OpenShift DR 集群和 hub 操作器的 ConfigMap 配置。

步骤

1. 在 Hub 集群中，导航到 **openshift-dr-system** 项目中的 Installed Operators，然后点击 **OpenShift DR Hub Operator**。您应该会看到两个可用的 API，DRPolicy 和 DRPlacementControl。
2. 为 DRPolicy 点 **Create instance** 并点击 **YAML 视图**。
3. 在将 *<cluster1>* 和 *<cluster2>* 替换为 ACM 中的受管集群的正确名称后，复制并保存到名为 **drpolicy.yaml** 的文件。使用任意值（只要它们是唯一的，例如 east 和 west）替换 *<string_value_1>* 和 *<string_value_2>*。**schedulingInterval** 应该是之前在 **MirrorPeer** 中配置的值之一（例如：5m)。

```
apiVersion: ramendr.openshift.io/v1alpha1
kind: DRPolicy
metadata:
  name: odr-policy-5m
spec:
  drClusterSet:
  - name: <cluster1>
    region: <string_value_1>
    s3ProfileName: s3profile-<cluster1>-ocs-storagecluster
  - name: <cluster2>
    region: <string_value_2>
    s3ProfileName: s3profile-<cluster2>-ocs-storagecluster
  schedulingInterval: 5m
```



注意

不需要指定一个命名空间来创建此资源，因为 DRPolicy 是一个集群范围的资源。

4. 将唯一 **drpolicy.yaml** 文件的内容复制到 YAML 视图中。您必须完全替换原始的内容。
5. 在 YAML 视图屏幕上点 **Create**。

重要

DRPolicy **schedulingInterval** ***必须***匹配在 MirroPeer 资源中配置的一个值（例如 5m）。要在 MirrorPeer 中配置的卷复制使用其他 DR **Intervals**，则需要使用新值（如 15m）创建额外的 DRPolicy 资源。确保将 DRPolicy **名称**更改为一个唯一值，这有助于标识复制间隔（如 odr-policy-15m）。

6. 通过在 **Hub 集群** 上运行命令来验证已成功为每个创建的 DRPolicy 资源创建了 DRPolicy ：这个示例是 **odr-policy-5m** ：

```
$ oc get drpolicy odr-policy-5m -n openshift-dr-system -o
jsonpath='{.status.conditions[].reason}{"\n"}'
```

输出示例：

```
Succeeded
```

第 9 章 启用自动安装 OPENSHIFT DR 集群 OPERATOR

在成功创建 DRPolicy 后，**OpenShift DR Cluster operator** 可以安装到主受管集群，并在 **openshift-dr-system** 命名空间中安装二级受管集群。

步骤

1. 编辑 Hub 集群上的 ConfigMag **ramen-hub-operator-config**，添加 **deploymentAutomationEnabled=true**，如下所示：

```
$ oc edit configmap ramen-hub-operator-config -n openshift-dr-system
```

```
apiVersion: v1
data:
  ramen_manager_config.yaml: |
    apiVersion: ramendr.openshift.io/v1alpha1
    drClusterOperator:
      deploymentAutomationEnabled: true  ## <-- Add to enable installation of ODR Cluster
operator on managed clusters
      catalogSourceName: redhat-operators
      catalogSourceNamespaceName: openshift-marketplace
      channelName: stable-4.10
      clusterServiceVersionName: odr-cluster-operator.v4.10.0
      namespaceName: openshift-dr-system
      packageName: odr-cluster-operator
[...]
```

2. 验证在**主受管集群**和**二级受管集群**中是否安装否成功，使用以下命令：

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

输出示例：

```
NAME                                                         DISPLAY                       VERSION
REPLACES   PHASE
clusterserviceversion.operators.coreos.com/odr-cluster-operator.v4.10.0   Openshift DR
Cluster Operator   4.10.0              Succeeded

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

您还可以进入每个受管集群的 OperatorHub，并确认是否安装了 **OpenShift DR Cluster Operator**。

第 10 章 启用自动向受管集群传输 S3SECRETS

按照以下步骤，将 s3Secrets 自动传送到所需的 OpenShift DR 集群组件。它使用访问 OpenShift DR 配置映射中 s3Profiles 所需的 s3Secrets 更新 OpenShift DR 集群命名空间。

流程

1. 编辑 Hub 集群上的 ConfigMag **ramen-hub-operator-config** 以添加 **s3SecretDistributionEnabled=true**，如下所示：

```
$ oc edit configmap ramen-hub-operator-config -n openshift-dr-system
```

```
apiVersion: v1
data:
  ramen_manager_config.yaml: |
    apiVersion: ramendr.openshift.io/v1alpha1
    drClusterOperator:
      deploymentAutomationEnabled: true
      s3SecretDistributionEnabled: true  ## <-- Add to enable automatic transfer of s3secrets
      catalogSourceName: redhat-operators
      catalogSourceNamespaceName: openshift-marketplace
      channelName: stable-4.10
      clusterServiceVersionName: odr-cluster-operator.v4.10.0
      namespaceName: openshift-dr-system
      packageName: odr-cluster-operator
[...]
```

2. 在两个受管集群中运行此命令，验证 secret 的传输是否成功。

```
$ oc get secrets -n openshift-dr-system | grep Opaque
```

输出示例：

```
8b3fb9ed90f66808d988c7edfa76eba35647092   Opaque       2      11m
af5f82f21f8f77faf3de2553e223b535002e480   Opaque       2      11m
```

第 11 章 创建示例应用程序

为了测试从主受管集群到二级受管集群的**故障切换**，我们需要一个简单的应用程序。使用名为 **busybox** 的示例应用作为示例。

步骤

1. 在 Hub 集群上为 **busybox** 示例应用程序创建一个**命名空间或项目**。

```
$ oc new-project busybox-sample
```

注意

如果需要，可以使用除 **busybox-sample** 以外的不同项目名称。在通过 Advanced Cluster Manager 控制台部署示例应用程序时，确保使用与此步骤中创建相同的项目名称。

2. 创建 **DRPlacementControl** 资源
   DRPlacementControl 是一个 API，在 Hub 集群上安装 OpenShift DR Hub Operator 后可用。它基本上是一个 Advanced Cluster Manager PlacementRule 协调器，它根据作为 DRPolicy 一部分的集群中的数据可用性来编排放置决策。

   a. 在 Hub 集群中，导航到 **busybox-sample** 项目中的 Installed Operators，再单击 **OpenShift DR Hub Operator**。您应该会看到两个可用的 API，DRPolicy 和 DRPlacementControl。

   b. 为 **DRPlacementControl** 创建一个实例，然后进入 YAML 视图。确保已选中 **busybox-sample** 项目。

   c. 在将 *<cluster1>* 替换为 Advanced Cluster Manager 中的受管集群的正确名称后，将以下 YAML 保存到名为 **busybox-drpc.yaml** 的文件。修改具有所需复制间隔的 **DRPolicy** 的 **drPolicyRef** *name*。

```
apiVersion: ramendr.openshift.io/v1alpha1
kind: DRPlacementControl
metadata:
  labels:
    app: busybox-sample
  name: busybox-drpc
spec:
  drPolicyRef:
    name: odr-policy-5m     ## <-- Modify to specify desired DRPolicy and RPO
  placementRef:
    kind: PlacementRule
    name: busybox-placement
  preferredCluster: <cluster1>
  pvcSelector:
    matchLabels:
      appname: busybox
```

   d. 将唯一 **busybox-drpc.yaml** 文件的内容复制到 YAML 视图（完全替换原始内容）。

   e. 在 YAML 视图屏幕上点 **Create**。
   您还可以使用以下 CLI 命令创建此资源：

```
$ oc create -f busybox-drpc.yaml -n busybox-sample
```

输出示例：

```
drplacementcontrol.ramendr.openshift.io/busybox-drpc created
```

重要

此资源必须在 **busybox-sample** 命名空间（或您之前创建的任何命名空间）中创建。

3. 创建 **Placement Rule** 资源，以定义可以部署资源模板的目标集群。使用放置规则促进应用程序的多集群部署。

   a. 将以下 YAML 复制到名为 **busybox-placementrule.yaml** 的文件中。

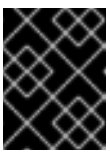
```
apiVersion: apps.open-cluster-management.io/v1
kind: PlacementRule
metadata:
  labels:
    app: busybox-sample
  name: busybox-placement
spec:
  clusterConditions:
  - status: "True"
    type: ManagedClusterConditionAvailable
  clusterReplicas: 1
  schedulerName: ramen
```

   b. 为 **busybox-sample** 应用创建 Placement Rule 资源。

```
$ oc create -f busybox-placementrule.yaml -n busybox-sample
```

输出示例：

```
placementrule.apps.open-cluster-management.io/busybox-placement created
```



重要

此资源必须在 **busybox-sample** 命名空间（或您之前创建的任何命名空间）中创建。

4. 使用 RHACM 控制台创建**示例应用程序**

   a. 如果尚未登录，请使用您的 OpenShift 凭据登录 RHACM 控制台。

```
$ oc get route multicloud-console -n open-cluster-management -o jsonpath --
template="https://{.spec.host}/multicloud/applications{"\n"}"
```

输出示例：

```
https://multicloud-console.apps.perf3.example.com/multicloud/applications
```

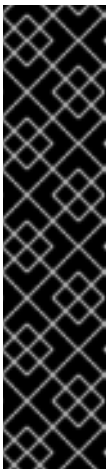
b. 导航到 **Applications**，再单击 **Create application**。

c. 选择类型 **Subscription**。

d. 输入应用程序 **Name** （如 **busybox**）和 **Namespace** （如 **busybox-sample**）。

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

f. 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/RamenDR/ocm-ramen-samples**，其中 **Branch** 是 **main**，**Path** 是 **busybox-odr**。



重要

在继续操作前，确保创建了新的 StorageClass **ocs-storagecluster-ceph-rbdmirror**，如创建 Mirror StorageClass 资源 部分所述。

使用以下命令验证是否已创建：

```
oc get storageclass | grep rbdmirror | awk '{print $1}'
```

输出示例：

```
ocs-storagecluster-ceph-rbdmirror
```

g. 将表单向下滚动到 **Select clusters to deploy** 部分，然后点击 **Select an existing placement configuration**。

h. 从下拉列表中选择 **现有 Placement Rule** （如 **busybox-placement**）。

i. 点击 **Save**。
在下一屏幕上，滚动到底部。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。

5. 验证示例应用部署和复制。
现在 **busybox** 应用程序已部署到首选的集群（在 DRPlacementControl 中指定），可以验证部署。

a. 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   1/1     Running   0          6m

NAME                                 STATUS   VOLUME                                     CAPACITY
```

```
ACCESS MODES   STORAGECLASS                 AGE
persistentvolumeclaim/busybox-pvc   Bound    pvc-a56c138a-a1a9-4465-927f-
af02afbbff37   1Gi        RWO           ocs-storagecluster-ceph-rbd   6m
```

b. 验证是否也为 **busybox** PVC 创建了复制资源。

```
$ oc get volumereplication,volumereplicationgroup -n busybox-sample
```

输出示例：

```
NAME                                                AGE   VOLUMEREPLICATIONCLASS
PVCNAME       DESIREDSTATE   CURRENTSTATE
volumereplication.replication.storage.openshift.io/busybox-pvc   6m   odf-rbd-
volumereplicationclass   busybox-pvc   primary        Primary

NAME                                          AGE
volumereplicationgroup.ramendr.openshift.io/busybox-drpc   6m
```

c. 通过对**主受管集群**和**次受管集群**运行以下命令，验证 **busybox** 卷是否已复制到备用集群。

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

输出示例：

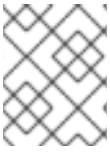
```
{"daemon_health":"OK","health":"OK","image_health":"OK","states":{"replaying":2}}
```

注意

两个受管集群都应该具有完全相同的输出，新的状态为 **"states": {"replaying":2}'**。

11.1. 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 应用时，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**。

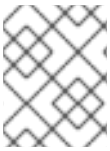
   a. 登录到 Hub 集群的 OpenShift Web 控制台，再进入为 **busybox-sample** 项目安装的 Operators。

   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。也可以使用 CLI，将 **DRPlacementControl** 资源删除到应用命名空间中。

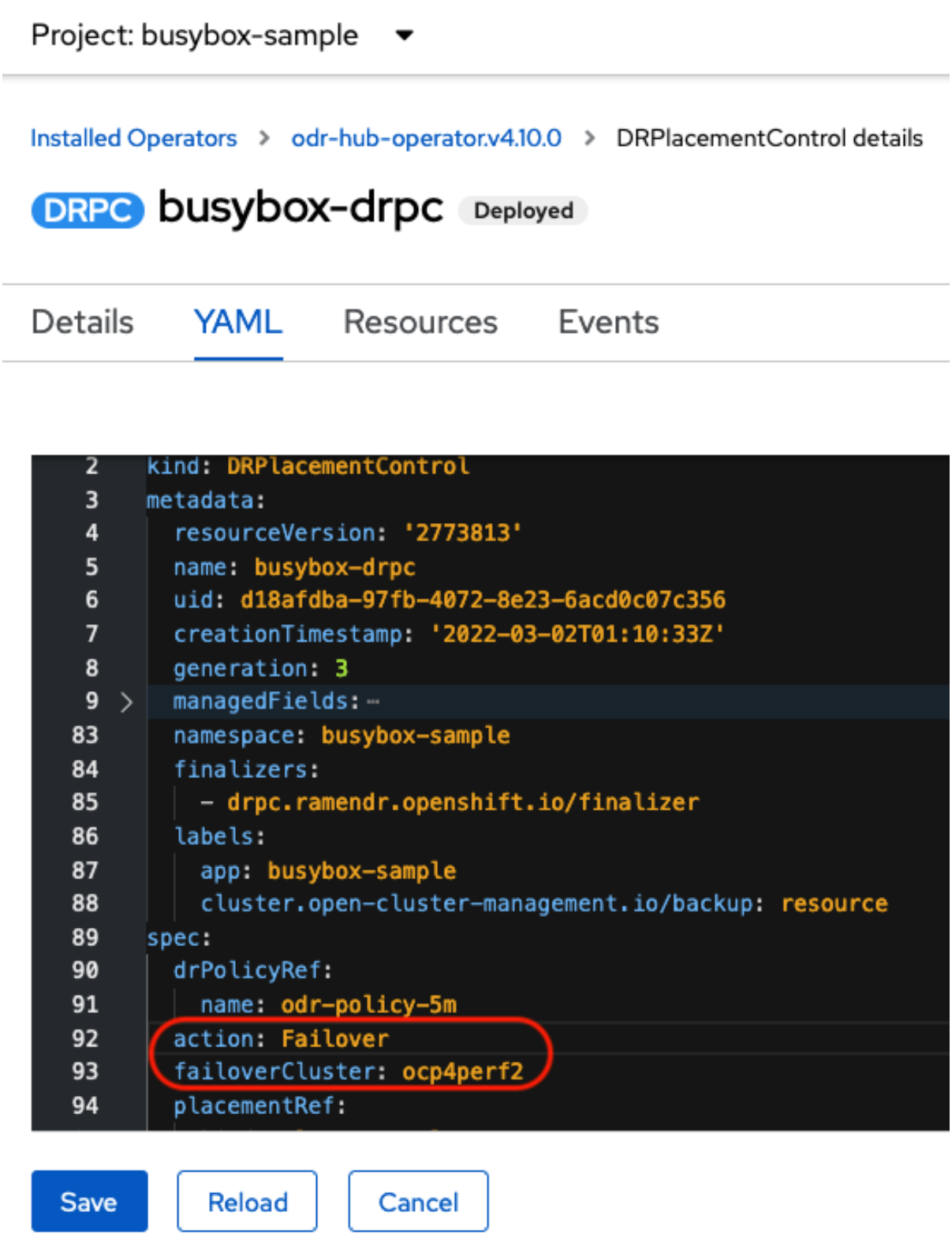
第 12 章 受管集群之间的应用程序故障切换

本节介绍如何故障转移 busybox 示例应用程序。区域DR 的故障转移方法基于应用程序。以这种方式保护的每个应用都必须具有对应的 **DRPlacementControl** 资源，并在应用程序**命名空间**中创建 **PlacementRule** 资源，如 Create Sample Application for DR 测试部分所示。

步骤

1. 在 Hub 集群中，导航到 Installed Operators，然后点 **Openshift DR Hub Operator**。
2. 单击 **DRPlacementControl** 选项卡。
3. 单击 DRPC **busybox-drpc**，然后单击 YAML 视图。
4. 添加 **action** 和 **failoverCluster** 详情，如下方屏幕截图所示。**failoverCluster** 应该是第二个受管集群的 ACM 集群名称。

   DRPlacementControl 添加 Failover 操作



5. 点击 **Save**。

6. 验证 application **busybox** 是否现在在次受管集群中运行，即 YAML 文件中指定的故障转移集群 **ocp4perf2**。

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

输出示例：

```
NAME          READY   STATUS    RESTARTS   AGE
pod/busybox   1/1     Running   0          35s
```

```
NAME                          STATUS   VOLUME                                     CAPACITY   ACCESS
MODES   STORAGECLASS                AGE
persistentvolumeclaim/busybox-pvc   Bound    pvc-79f2a74d-6e2c-48fb-9ed9-666b74cfa1bb
5Gi        RWO            ocs-storagecluster-ceph-rbd   35s
```

7. 验证 **busybox** 不再在主受管集群上运行。

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

输出示例：

```
No resources found in busybox-sample namespace.
```

重要

请注意发行说明的已知区域DR 问题，如已知问题部分所述。

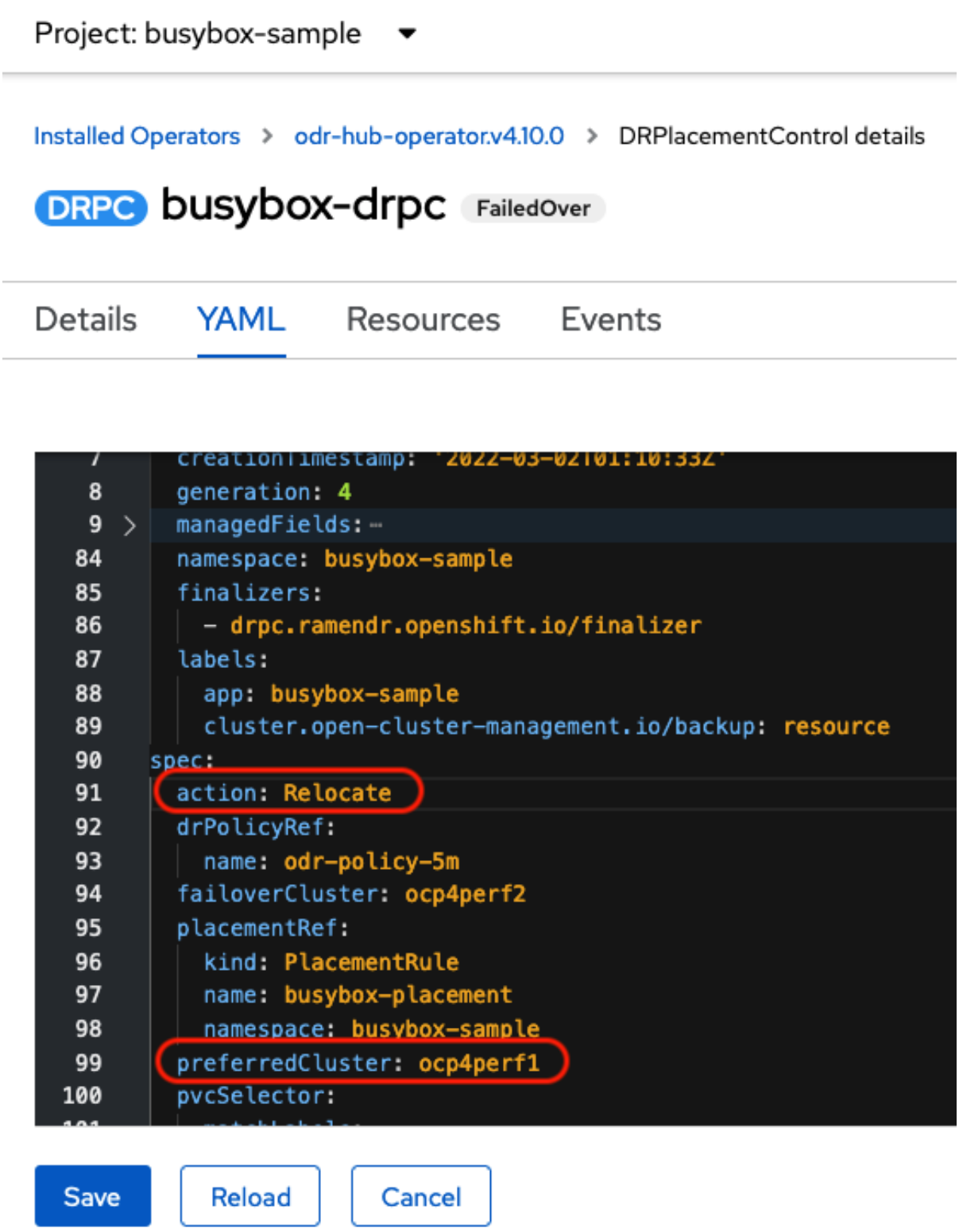
第 13 章 在受管集群间重新定位应用程序

重定位操作与故障转移非常相似。重新定位基于应用，并使用 DRPlacementControl 来触发重定位。重定位的主要区别在于，会发出 **重新同步** 以确保第二受管集群上保存的任何新应用程序数据都会立即出现，而不会等待镜像调度间隔复制到主要受管集群。

步骤

1. 在 Hub 集群中，导航到 Installed Operators，然后点 **Openshift DR Hub Operator**。
2. 单击 **DRPlacementControl** 选项卡。
3. 单击 DRPC **busybox-drpc**，然后单击 YAML 视图。
4. 将**操作**改为 **Relocate**

   DRPlacementControl 修改重新分配的操作



5. 点击 **Save**。

6. 验证 **busybox** 应用程序已在主受管集群中运行。故障转移是在 YAML 文件中指定的 **preferredCluster ocp4perf1**，它是故障转移操作前应用程序运行的位置。

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

输出示例：

```
NAME          READY   STATUS    RESTARTS   AGE
pod/busybox   1/1     Running   0          60s
```

```
NAME                          STATUS   VOLUME                                     CAPACITY   ACCESS
MODES   STORAGECLASS                AGE
persistentvolumeclaim/busybox-pvc   Bound    pvc-79f2a74d-6e2c-48fb-9ed9-666b74cfa1bb
5Gi        RWO            ocs-storagecluster-ceph-rbd   61s
```

7. 验证 **busybox** 是否在第二个受管集群中运行。busybox 应用程序不应在此受管集群上运行。

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

输出示例：

```
No resources found in busybox-sample namespace.
```

重要

请注意发行说明的已知区域DR 问题，如已知问题部分所述。