https://red-hat-storage.github.io/ocs-training/training/ocs4/odf410-multisite-ramen.html

Disaster recovery is the ability to recover and continue business critical applications from natural or human created disasters. It is the overall business continuance strategy of any major organization as designed to preserve the continuity of business operations during major adverse events.

Regional-DR capability provides volume persistent data and metadata replication across sites that are geographically dispersed. In the public cloud these would be similar to protecting from a region failure. Regional-DR ensures business continuity during the unavailability of a geographical region, accepting some loss of data in a predictable amount. This is usually expressed at Recovery Point Objective (RPO) and Recovery Time Objective (RTO).

RPO is a measure of how frequently you take backups or snapshots of persistent data. In practice, the RPO indicates the amount of data that will be lost or need to be reentered after an outage.

RTO is the amount of downtime a business can tolerate. The RTO answers the question, “How long can it take for our system to recover after we were notified of a business disruption?”

The intent of this guide is to detail the steps and commands necessary to be able to failover an application from one OpenShift Container Platform (OCP) cluster to another and then failback the same application to the original primary cluster. In this case the OCP clusters will be created or imported using Red Hat Advanced Cluster Management or RHACM.

This is a general overview of the steps required to configure and execute OpenShift Disaster Recovery or ODR capabilities using OpenShift Data Foundation (ODF) v4.10 or higher and RHACM v2.4 or higher across two distinct OCP clusters separated by distance. In addition to these two cluster called managed clusters, there is currently a requirement to have a third OCP cluster that will be the Advanced Cluster Management (ACM) hub cluster.

Components of Regional-DR Solution

Regional-DR is composed of Red Hat Advanced Cluster Management for Kubernetes (RHACM) and OpenShift Data Foundation components to provide application and data mobility across OpenShift Container Platform clusters.

Red Hat Advanced Cluster Management for Kubernetes (RHACM)

RHACM provides the ability to manage multiple clusters and application lifecycles. Hence, it serves as a control plane in a multi-cluster environment.

RHACM is split into two parts:

- RHACM Hub: includes component that run on the multi-cluster control plane.
- Managed clusters: includes components that run on the clusters that are managed.

OpenShift Data Foundation

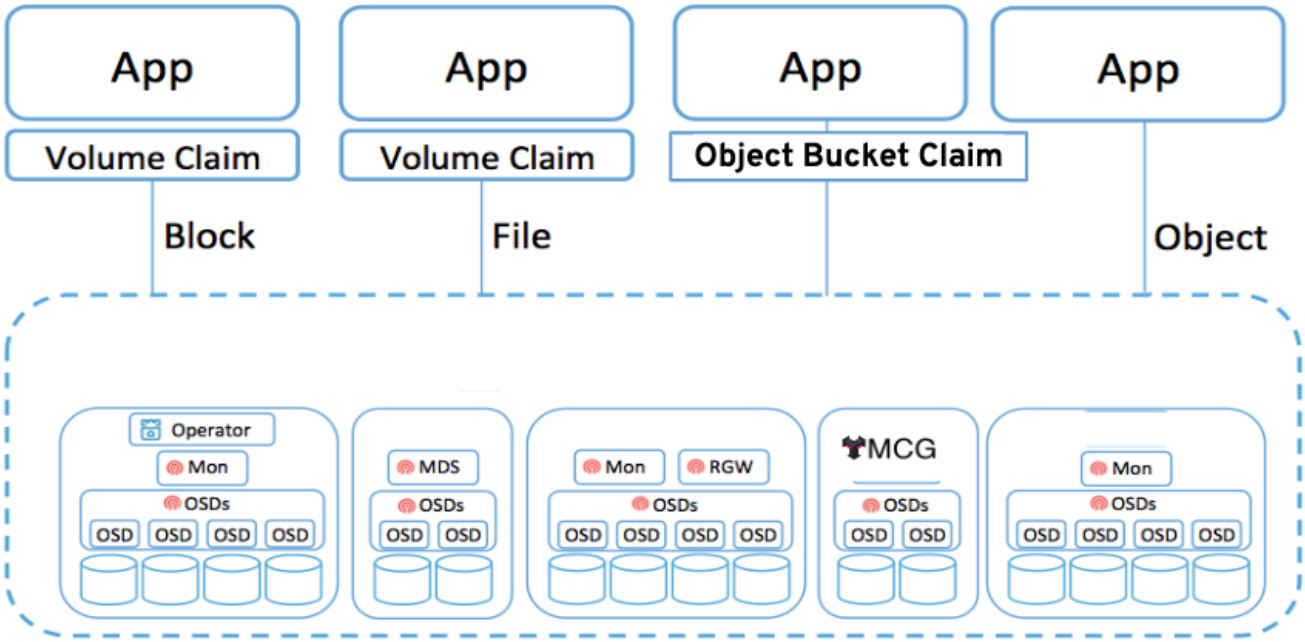
OpenShift Data Foundation provides the ability to provision and manage storage for stateful applications in an OpenShift Container Platform cluster.

OpenShift Data Foundation is backed by Ceph as the storage provider, whose lifecycle is managed by Rook in the OpenShift Data Foundation component stack.

Ceph-CSI provides the provisioning and management of Persistent Volumes for stateful applications.

OpenShift Data Foundation stack is now enhanced with the following abilities:

- Enable pools for mirroring
- Automatically mirror images across RBD block pools
- Provides csi-addons to manage per Persistent Volume Claim (PVC) mirroring <<<



Openshift DR

OpenShift DR is a disaster recovery orchestrator for stateful applications across a set of peer OpenShift clusters which are deployed and managed using RHACM and provides cloud-native interfaces to orchestrate the life-cycle of an application's state on Persistent Volumes.

These include:

- Protecting an application state relationship across OpenShift clusters
- Failing over an application's state to a peer cluster
- Relocate an application's state to the previously deployed cluster

OpenShift DR is split into three components:

1. ODF Multicluster Orchestrator:

- Installed on the multi-cluster control plane (RHACM Hub), it also performs the following actions:
- Creates a bootstrap token and exchanges this token between the managed clusters.
- Enables mirroring for the default CephBlockPool on the managed clusters.
- Creates an object bucket using Multicloud Object Gateway (MCG) on each managed cluster for PVC and PV metadata.
- Creates a Secret for each new object bucket that has the keys for bucket access on the Hub cluster in the openshift-dr-system project.

- Creates a VolumeReplicationClass on the Primary managed cluster and the Secondary managed cluster for each schedulingIntervals (e.g. 5m, 15m, 30m).
- Modifies the ramen-hub-operator-config ConfigMap on the Hub cluster and adds the s3StoreProfiles entries.

2. OpenShift DR Hub Operator:

- Installed on the hub cluster to manage failover and relocation for applications.

3. OpenShift DR Cluster Operator:

- Installed on each managed cluster to manage the lifecycle of all PVCs of an application.

Regional-DR deployment workflow - Configure and Execute Regional DR Capabilities

Deploy and Configure ACM for Multisite connectivity

1. Install the ACM operator on the hub cluster. x After creating the OCP hub cluster, install from OperatorHub the ACM operator. After the operator and associated pods are running, create the MultiClusterHub resource.

2. Create or import managed OCP clusters into ACM hub. Import or create the two managed clusters with adequate resources for ODF (compute nodes, memory, cpu) using the RHACM console. You need to create a cluster set named "manage-cluster-set" and add both these clusters to this cluster set in ACM.

3. Ensure clusters have unique private network address ranges.

   IMPORTANT: Ensure the primary and secondary OCP clusters have unique private network address ranges (i.e., `clusterNetwork` and `serviceNetwork`). Required for Submariner add-on `oc get networks.config.openshift.io cluster -o json | jq .spec`

4. Connect the private networks using Submariner add-ons. Connect the managed OCP private networks (cluster and service) using the RHACM Submariner add-ons.

   Submariner is an open-source tool that can be used to provide direct networking between two or more Kubernetes clusters in a given ManagedClusterSet, either on-premises or in the cloud.

OpenShift Data Foundation Installation

In order to configure storage replication between the two OCP clusters OpenShift Data Foundation (ODF) must be installed first on each managed cluster. ODF deployment guides and instructions are specific to your infrastructure (i.e. Openstack, AWS, VMware, BM, Azure, etc.). Install ODF version 4.10 or greater on both OCP managed clusters.

OpenShift Data Foundation can be deployed either entirely within OpenShift Container Platform (internal approach) or where the services run from a cluster outside OpenShift Container Platform (external approach).

Internal Approach

- Deployment of Red Hat OpenShift Data Foundation entirely within Red Hat OpenShift Container Platform provides the benefits of operator-based deployment and management. There are two deployment modalities available—simple or optimized—when Red Hat OpenShift Data Foundation is running entirely within Red Hat OpenShift Container Platform.

Simple Deployment: Red Hat OpenShift Data Foundation services run coresident with applications and are managed by operators in Red Hat OpenShift Container Platform.

A simple deployment is best for the following situations:

- Storage requirements are not well defined.
- OpenShift Data Foundation services are to run coresident with applications.
- It is difficult to create a node instance of a specific size (bare metal).

For Red Hat OpenShift Data Foundation to run coresident with applications, they must have local storage devices, or portable storage devices attached to them dynamically. Examples are EBS volumes on EC2 or vSphere Virtual Volumes on VMware.

Optimized Deployment: OpenShift Data Foundation services run on dedicated infrastructure nodes managed by Red Hat OpenShift Container Platform.

An optimized approach is best for the following types of situations:

- Storage requirements are well defined.
- OpenShift Data Foundation services run on dedicated infrastructure nodes.
- It is easy to create a node instance of a specific size—for example, for a cloud or virtualized environment.

External Approach

- Red Hat OpenShift Data Foundation exposes the services of Red Hat Ceph Storage running outside the OpenShift Container Platform cluster as storage classes.

The external approach is well suited to the following situations:

- Storage requirements are significant (600-plus storage devices).
- Multiple OpenShift Container Platform clusters need to consume storage services from a common external cluster.
- Another team (such as SRE or Storage) needs to manage the external cluster providing storage services—possibly pre-existing.

After installing the OCS operator, you can create a storage cluster resource from OpenShift Web Console using an installation wizard, or submitted manually—in either internal or external mode.

5. Install ODF 4.10 on managed clusters.* Install ODF 4.10 on primary and secondary OCP managed clusters and validate deployment.

   ```
   oc –n openshift–storage get pods
   ```

   You can validate the successful deployment of ODF on each managed OCP cluster with the following command:

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

And for the Multi-Cluster Gateway (MCG):

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

6. Install ODR Hub Operator on the ACM hub cluster. Install from OperatorHub on the ACM hub cluster the ODR Hub Operator.

Configure Multisite Storage Replication

Mirroring or replication is enabled on a per CephBlockPool basis within peer managed clusters and can then be configured on a specific subset of images within the pool. The rbd-mirror daemon is responsible for replicating image updates from the local peer cluster to the same image in the remote cluster.

7. Install ODF Multicluster Orchestrator operator on ACM hub cluster. Using OperatorHub on ACM hub cluster install the multicluster orchestrator operator.

   Orchestrator for OpenShift Data Foundation clusters running across multiple OpenShift clusters. It uses Red Hat Advanced Cluster Management for Kubernetes as the multicluster control plane.

Create Mirror Peer on Hub cluster

`Mirror Peer` is a cluster-scoped resource to hold information about the managed clusters that will have a peering relationship.

The job of the `ODF Multicluster Orchestrator` controller and the `MirrorPeer` Custom Resource, is to do the following:

- Create a bootstrap token and exchanges this token between the managed clusters.
- Enable mirroring for the default CephBlockPool on each managed clusters.
- Create an object bucket (using MCG) on each managed cluster for mirrored PVC and PV metadata.
- Create a Secret in the openshift-dr-system project on the Hub cluster for each new object bucket that has the base64 encoded access keys.
- Create a VolumeReplicationClass on the Primary managed cluster and the Secondary managed cluster for each schedulingIntervals (e.g. 5m, 15m, 30m).
- Modify the ramen-hub-operator-config ConfigMap on the Hub cluster and add the s3StoreProfiles entries.

Requirements:

- Must be installed on Hub cluster after the ODF Multicluster Orchestrator is installed on Hub cluster.
- There can only be two clusters per Mirror Peer.
- Each cluster should be uniquely identifiable in ACM by cluster name (i.e., cluster1).

8. Install Mirror Peer resource on ACM hub cluster. Using the multicluster orchestrator operator, install the MirrorPeer resource using CLI or the operator wizard.

```
cat <<EOF | kubectl apply -f -
apiVersion: multicluster.odf.openshift.io/v1alpha1
kind: MirrorPeer
metadata:
  name: mirrorpeer-primary-secondary
spec:
  items:
  - clusterName: primary-cluster
    storageClusterRef:
      name: ocs-storagecluster
      namespace: openshift-storage
  - clusterName: secondary-cluster
    storageClusterRef:
      name: ocs-storagecluster
      namespace: openshift-storage
  manageS3: true
  schedulingIntervals:
  - 5m
  - 15m
EOF
```

9. Validate Ceph mirroring is active between managed OCP clusters Validate mirroring is enabled on default CephBlockPool on the Primary managed cluster and the Secondary managed cluster. This may take about 5 minutes before the peer mirroring is established.

   Using CLI, validate the new rbd-mirroring pods are created in each managed cluster and that the default CephBlockPool has healthy mirroring status in both directions.

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

   Validate the status of the daemon health on the Primary managed cluster and the Secondary managed cluster.

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

   > It could take up to 10 to 15 minutes for the daemon_health and health to go from Warning to OK. If the status does not become OK eventually then use the ACM console to verify that the Submariner add-ons connection is still in a healthy state.

   Validate that a VolumeReplicationClass is created on the Primary managed cluster and the Secondary managed cluster for each schedulingIntervals listed in the MirrorPeer (e.g. 5m, 15m).

   ```
   oc get volumereplicationclass
   ```

   > The VolumeReplicationClass is used to specify the mirroringMode for each volume to be replicated as well as how often a volume or image is replicated (for example, every 5 minutes) from the local cluster to the remote cluster.

10. Validate Object Buckets and s3StoreProfiles. Using CLI, validate that new MCG object buckets are created on the managed clusters including new secrets stored in the hub cluster and s3StoreProfiles added to the ramen config map.

    > The new object buckets are created in the openshift-storage namespace on the managed clusters.

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

    Validate there are two new Secrets in the Hub cluster openshift-dr-system namespace that contain the access and secret key for each new OBC.

    > The new secrets are created in the openshift-dr-system namespace on the Hub cluster. Later, these secrets will be copied to the managed clusters to access the object buckets.

    The OBC and Secrets are written in the ConfigMap ramen-hub-operator-config on the Hub cluster in the newly created s3StoreProfiles section.

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

    > Record the names of the s3ProfileName. They will be used in the DRPolicy resource.

Create new mirroring StorageClass resource.

The block volumes with mirroring enabled must be created using a new StorageClass that has additional imageFeatures required to enable faster image replication between managed clusters. The new features are exclusive-lock, object-map, and fast-diff. The default ODF StorageClass ocs-storagecluster-ceph-rbd does not include these features.

11. Create new mirroring StorageClass resource. Using CLI, create new StorageClass with correct image features for block volumes enabled for mirroring.

> This resource must be created on the Primary managed cluster and the Secondary managed cluster.

```
cat <<EOF | kubectl apply -f -
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
EOF
```

12. Configure SSL access between managed clusters if (if needed). For each managed cluster extract the ingress certificate and inject into the alternate cluster for MCG object bucket secure access.

These steps are necessary so that metadata can be stored on the alternate cluster in a Multi-Cloud Gateway (MCG) object bucket using a secure transport protocol and in addition the Hub cluster needs to verify access to the object buckets.

Extract the ingress certificate for the Primary managed cluster and save the output to primary.crt.

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

Extract the ingress certificate for the Secondary managed cluster and save the output to secondary.crt.

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

Create a new YAML file cm-clusters-crt.yaml to hold the certificate bundle for both the Primary managed cluster and the Secondary managed cluster.

```
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
```

> This ConfigMap needs to be created on the Primary managed cluster, Secondary managed cluster, and the Hub cluster.

After all the user-ca-bundle ConfigMaps are created, the default Proxy cluster resource needs to be modified.

Patch the default Proxy resource on the Primary managed cluster, Secondary managed cluster, and the Hub cluster.

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

Create the DRPolicy resource on the hub cluster

ODR uses the DRPolicy resources on the ACM hub cluster to deploy, failover, and relocate, workloads across managed clusters. A DRPolicy requires a set of two clusters, which are peered for storage level replication and CSI VolumeReplication is enabled.

Furthermore, DRPolicy requires a scheduling interval that determines at what frequency data replication will be performed and also serves as a coarse grained RPO (Recovery Point Objective) for the workload using the DRPolicy.

DRPolicy also requires that each cluster in the policy be assigned a S3 profile name, which is configured via the ConfigMap ramen-hub-operator-config in the openshift-dr-system on the Hub cluster.

On the Hub cluster navigate to Installed Operators in the openshift-dr-system project and select ODR Hub Operator. You should see two available APIs, DRPolicy and DRPlacementControl.

13. Create the DRPolicy resource on the hub cluster. DRPolicy is an API available after the ODR Hub Operator is installed. It is used to deploy, failover, and relocate, workloads across managed clusters.

```
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
```

> The DRPolicy schedulingInterval must match one of the values configured in MirroPeer resource (e.g. 5m). To use one of the other schedulingIntervals for volume replication configured in the MirrorPeer requires creating additional DRPolicy resources with the new values (i.e., 15m). Make sure to change the DRPolicy name to be unique and useful in identifying the replication interval (e.g. odr-policy-15m).

14. Enable Automatic Install of ODR Cluster operator. Enable the ODR Cluster operator to be installed from the hub cluster to the managed cluster by setting deploymentAutomationEnabled=true in the ramen config map.

    Once the DRPolicy is created successfully the ODR Cluster operator can be installed on the Primary managed cluster and Secondary managed cluster in the openshift-dr-system namespace.

    This is done by editing the ramen-hub-operator-config ConfigMap on the Hub cluster and adding deploymentAutomationEnabled=true.

    ```
    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
    ```

    To validate that the installation was successful on the Primary managed cluster and the Secondary managed cluster do the following command:

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

15. Create S3 secrets on managed clusters. Copy the S3 secrets on hub cluster to YAML files. Create both secrets on the managed clusters.

    The MCG object bucket Secrets were created and stored on the Hub cluster when the MirrorPeer was created.

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

These Secrets need to be copied to the Primary managed cluster and the Secondary managed cluster. An easy way to do this is to export each Secret to a YAML file on the Hub cluster. Here is an example using the Secret names in the Example output above.

```
oc get secrets 8b3fb9ed90f66808d988c7edfa76eba35647092 -n openshift-dr-system -o yaml > odr-s3-secret1.yaml
```

```
oc get secrets af5f82f21f8f77faf3de2553e223b535002e480 -n openshift-dr-system -o yaml > odr-s3-secret2.yaml
```

Now create both these Secrets on the Primary managed cluster and the Secondary managed cluster. They are created in the openshift-dr-system namespace.

Create the Sample Application namespace on the hub cluster.

In order to test failover from the Primary managed cluster to the Secondary managed cluster and back again we need a simple application. The sample application used for this example with be busybox.

16. Create the Sample Application namespace on the hub cluster. Because the ODR Hub Operator APIs are namespace scoped, the sample application namespace must be created first.

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

17. Create the DRPlacementControl resource on the hub cluster. DRPlacementControl is an API available after the ODR Hub Operator is installed.

    DRPlacementControl is an API available after the ODR Hub Operator is installed on the Hub cluster. It is broadly an ACM PlacementRule reconciler that orchestrates placement decisions based on data availability across clusters that are part of a DRPolicy.

    On the Hub cluster navigate to Installed Operators in the busybox-sample project and select ODR Hub Operator. You should see two available APIs, DRPolicy and DRPlacementControl.

    Create instance for DRPlacementControl and then go to YAML view. Make sure the busybox-sample namespace is selected at the top.

    Save the following YAML (below) to filename busybox-drpc.yaml after replacing with the correct name of your managed cluster in ACM. Modify drPolicyRef name for the DRPolicy that has the desired replication interval.

    ```
    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
```

Now create the DRPlacementControl resource by copying the contents of your unique busybox-drpc.yaml file into the YAML view (completely replacing original content). Select Create at the bottom of the YAML view screen.

18. Create the PlacementRule resource on the hub cluster. Placement rules define the target clusters where resource templates can be deployed.

    Placement rules define the target clusters where resource templates can be deployed. Use placement rules to help you facilitate the multicluster deployment of your applications.

```
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
```

19. Create the Sample Application using ACM console. Use the sample app example from https://github.com/RamenDR/ocm-ramen-samples to create a busybox deployment for failover and failback testing.

    After logging in select Create application in the top right and choose Subscription.

    The next section to fill out is below the Git box and is the repository URL for the sample application, the github branch and path to resources that will be created, the busybox Pod and PVC.

    Sample application repository https://github.com/RamenDR/ocm-ramen-samples. Branch is main and path is busybox-odr.

20. Validate Sample Application deployment and alternate cluster replication Using CLI commands on both managed clusters validate that the application is running and that the volume was replicated to the alternate cluster.

    Now that the busybox application has been deployed to your preferredCluster (specified in the DRPlacementControl) the deployment can be validated.

Logon to your managed cluster where busybox was deployed by ACM. This is most likely your Primary managed cluster.

To validate that the replication resources are also created for the busybox PVC do the following:

```
oc get volumereplication,volumereplicationgroup -n busybox-Sample
```

To validate that the busybox volume has been replicated to the alternate cluster run this command on both the Primary managed cluster and the Secondary managed cluster.

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

21. Deleting the Sample Application

Deleting the busybox application can be done using the ACM console. Navigate to Applications and then find the application to be deleted (busybox in this case).

Application Failover between managed clusters

This section will detail how to failover the busybox sample application. The failover method for Regional Disaster Recovery is application based. Each application that is to be protected in this manner must have a corresponding DRPlacementControl resource and a PlacementRule resource created in the application namespace as shown in the [Create Sample Application for DR testing] section.

Select drpc-busybox and then the YAML view. Add the action and failoverCluster as shown below. The failoverCluster should be the ACM cluster name for the Secondary managed cluster.

action: Failover failoverCluster: cluster2

21. Failover Sample Application to secondary managed cluster. Using the application DRPlacementControl resource on the Hub Cluster, add the action of Failover and specify the failoverCluster to trigger the failover.

Application Failback between managed clusters

A failback operation is very similar to failover. The failback is application based and uses the DRPlacementControl to trigger the failback. The main difference for failback is that a resync is issued to make sure any new application data saved on the Secondary managed cluster is immediately, not waiting for the mirroring schedule interval, replicated to the Primary managed cluster.

22. Failback Sample Application to primary managed cluster. Using the application DRPlacementControl resource on the Hub Cluster, modify the action to Relocate and trigger failover to the preferredCluster.

Deploy and Configure ACM for Multisite connectivity

This installation method requires you have three OpenShift clusters that have network reachability between them. For the purposes of this document we will use this reference for the clusters:

- Hub cluster is where ACM, ODF Multisite-orchestrator and ODR Hub controllers are installed on ODF Disaster Recovery Management cluster.

- Primary managed cluster is where ODF, ODR Cluster controller, and Applications are installed on ODF Disaster Recovery Site 1 cluster.
- Secondary managed cluster is where ODF, ODR Cluster controller, and Applications are installed on ODF Disaster Recovery Site 2 cluster.

Install ODR Hub Operator on the ACM hub cluster.

Create new CSI side-cars in each managed OCP cluster. Install ODF Multicluster-orchestrator operator on ACM hub cluster.

Install Mirror Peer resource on ACM hub cluster.* Enable Ceph mirroring on managed OCP clusters. Validate Ceph mirroring is active between managed OCP clusters. Create VolumeReplicationClass resource.* Create new mirroring StorageClass resource. Install ODR Cluster Operator on managed clusters. Configure SSL access between managed clusters. Configure S3 secrets and ODR configmaps for managed clusters. Install ODR Hub Operator on the hub cluster. Configure S3 secrets and ODR configmap for hub cluster. Create the DRPolicy resource on the hub cluster. Create the Sample Application namespace on the hub cluster. Create the DRPlacementControl resource on the hub cluster. Create the PlacementRule resource on the hub cluster. Create the Sample Application using ACM console. Validate Sample Application deployment and alternate cluster replication. Failover Sample Application to secondary managed cluster. Failback Sample Application to primary managed cluster.