Red Hat OpenShift Data Foundation 4.15

Deploying multiple OpenShift Data Foundation storage clusters

Instructions on how to deploy an external OpenShift Data Foundation storage cluster when you already have an existing OpenShift Data Foundation internal storage cluster.

Red Hat OpenShift Data Foundation 4.15 Deploying multiple OpenShift Data Foundation storage clusters

Instructions on how to deploy an external OpenShift Data Foundation storage cluster when you already have an existing OpenShift Data Foundation internal storage cluster.

Legal Notice

Abstract

Read this document for instructions about how to deploy an external OpenShift Data Foundation storage cluster when you already have an existing OpenShift Data Foundation internal storage cluster.

Table of Contents

MAKING OPEN SOURCE MORE INCLUSIVE

Red Hat is committed to replacing problematic language in our code, documentation, and web properties. We are beginning with these four terms: master, slave, blacklist, and whitelist. Because of the enormity of this endeavor, these changes will be implemented gradually over several upcoming releases. For more details, see our CTO Chris Wright's message .

PROVIDING FEEDBACK ON RED HAT DOCUMENTATION

We appreciate your input on our documentation. Do let us know how we can make it better.

To give feedback, create a Bugzilla ticket:

1. Go to the Bugzilla website.

2. In the **Component** section, choose **documentation**.

3. Fill in the **Description** field with your suggestion for improvement. Include a link to the relevant part(s) of documentation.

4. Click **Submit Bug**.

CHAPTER 1. OVERVIEW OF MULTIPLE STORAGE CLUSTER DEPLOYMENTS

Red Hat OpenShift Data Foundation provides the ability to deploy two storage clusters, where one is internal mode and the other is in external mode. This can be achieved only with the first cluster installed as internal in **openshift-storage** namespace while the second cluster is installed as the external in **openshift-storage-extended** namespace. Clusters installed conversely is not currently supported.

Supported platforms

- Bare metal

- VMware VSphere

- OpenStack

- OpenShift Virtualization

- IBM Cloud

- IBM Power

CHAPTER 2. PREPARING TO DEPLOY MULTIPLE OPENSHIFT DATA FOUNDATION STORAGE CLUSTERS

Before you begin the deployment of OpenShift Data Foundation using dynamic, local, or external storage, ensure that your resource requirements are met. See the Resource requirements section in the Planning guide.

Things you should remember before installing multiple OpenShift Data Foundation storage clusters:

- **openshift-storage** and **openshift-storage-extended** are the exclusively supported namespaces.

- Internal storage cluster is restricted to the OpenShift Data Foundation operator namespace.

- External storage cluster is permissible in both operator and non-operator namespaces.

- Multiple storage clusters are not supported in the same namespace. Hence, the external storage system will not be visible under the OpenShift Data Foundation operator page as the operator is under **openshift-storage** namespace and the external storage system is not.

- Customers running external storage clusters in the operator namespace cannot utilize multiple storage clusters.

- **Multicloud Object Gateway** is supported solely within the operator namespace. It is ignored in other namespaces.

- **RADOS Gateway (RGW)** can be in either the operator namespace, a non-operator namespace, or both

- **Network File System (NFS)** is enabled as long as it is enabled for at least one of the clusters.

- Topology is enabled as long as it is enabled for at least one of the clusters.

- Topology domain labels are set as long as the internal cluster is present.

- The **Topology** view of the cluster is only supported for OpenShift Data Foundation internal mode deployments.

- Different multus settings are not supported for multiple storage clusters.

CHAPTER 3. DEPLOYING OPENSHIFT DATA FOUNDATION INTERNAL STORAGE CLUSTER

To deploy and verify OpenShift Data Foundation storage cluster in the internal mode, refer to your infrastructure specific deployment guides.

CHAPTER 4. DEPLOYING OPENSHIFT DATA FOUNDATION EXTERNAL STORAGE CLUSTER

Use this procedure to deploy an external storage cluster to add additional storage or expand your current internal storage cluster.

Prerequisites

- An OpenShift Data Foundation cluster deployed in internal mode.

- Ensure that both the OpenShift container Platform and OpenShift Data Foundation are upgraded to version 4.15.

Procedure

1. In the OpenShift web Console, navigate to **Storage → Data Foundation → Storage Systems** tab

2. Click **Create StorageSystem**.

3. In the Backing storage page, **Connect an external storage platform** is selected by default.

   a. Choose **Red Hat Ceph Storage** as the **Storage platform** from available options.

   b. Click **Next**.

4. In the Security and Network page,

   a. Optional: To select encryption, select **Enable encryption** checkbox.

   b. In the Connection section, click on the **Download Script** link to download the python script for extracting Ceph cluster details.

   c. For extracting the Red Hat Ceph Storage (RHCS) cluster details, run the downloaded python script on a Red Hat Ceph Storage node with the **admin key**.

      i. Run the following command on the RHCS node to view the list of available arguments:

      ```
      # python3 ceph-external-cluster-details-exporter.py --help
      ```

      You can also run the script from inside a MON container (containerized deployment) or from a MON node (RPM deployment).

      > **NOTE**
      >
      > Use the **yum install cephadm** command and then the **cephadm** command to deploy your RHCS cluster using containers. You must pull the RHCS container images using the **cephadm** command, rather than using **yum** for installing the Ceph packages onto nodes. For more information, see RHCS product documentation.
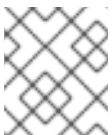
      ii. To retrieve the external cluster details from the RHCS cluster, run the following command:

```
# python3 ceph-external-cluster-details-exporter.py \
--rbd-data-pool-name <rbd block pool name>  [optional arguments]
```

For example:

```
# python3 ceph-external-cluster-details-exporter.py --rbd-data-pool-name ceph-rbd --
monitoring-endpoint xxx.xxx.xxx.xxx --monitoring-endpoint-port xxxx --rgw-endpoint
xxx.xxx.xxx.xxx:xxxx --run-as-user client.ocs
```

In this example,

rbd-data-pool-name	A mandatory parameter that is used for providing block storage in OpenShift Data Foundation.
rgw-endpoint	(Optional) This parameter is required only if the object storage is to be provisioned through Ceph Rados Gateway for OpenShift Data Foundation. Provide the endpoint in the following format: **<ip_address>:<port>** **NOTE** A fully-qualified domain name (FQDN) is also supported in the format **<FQDN>:<PORT>**.
monitoring-endpoint	(Optional) This parameter accepts comma-separated list of IP addresses of active and standby **mgrs** reachable from the OpenShift Container Platform cluster. If not provided, the value is automatically populated.
monitoring-endpoint-port	(Optional) It is the port associated with the **ceph-mgr** Prometheus exporter specified by **--monitoring-endpoint**. If not provided, the value is automatically populated.
run-as-user	(Optional) This parameter is used for providing name for the Ceph user which is created by the script. If this parameter is not specified, a default user name **client.healthchecker** is created. The permissions for the new user is set as: • caps: [mgr] allow command config • caps: [mon] allow r, allow command quorum_status, allow command version • caps: [osd] allow rwx pool=**RGW_POOL_PREFIX.rgw.meta**, allow r pool=**.rgw.root**, allow rw pool=**RGW_POOL_PREFIX.rgw.control**, allow rx pool=**RGW_POOL_PREFIX.rgw.log**, allow x pool=**RGW_POOL_PREFIX.rgw.buckets.index**

Additional flags:

rgw-pool-prefix	(Optional) The prefix of the RGW pools. If not specified, the default prefix is **default**.

rgw-tls-cert-path	(Optional) The file path of the RADOS Gateway endpoint TLS certificate.
rgw-skip-tls	(Optional) This parameter ignores the TLS certification validation when a self-signed certificate is provided (NOT RECOMMENDED).
ceph-conf	(Optional) The name of the Ceph configuration file.
cluster-name	(Optional) The Ceph cluster name.
output	(Optional) The file where the output is required to be stored.
cephfs-metadata-pool-name	(Optional) The name of the CephFS metadata pool.
cephfs-data-pool-name	(Optional) The name of the CephFS data pool.
cephfs-filesystem-name	(Optional) The name of the CephFS filesystem.
rbd-metadata-ec-pool-name	(Optional) The name of the erasure coded RBD metadata pool.
dry-run	(Optional) This parameter helps to print the executed commands without running them.
restricted-auth-permission	(Optional) This parameter restricts **cephCSIKeyrings** auth permissions to specific pools and clusters. Mandatory flags that need to be set with this are **rbd-data-pool-name** and **cluster-name**. You can also pass the **cephfs-filesystem-name** flag if there is CephFS user restriction so that permission is restricted to a particular CephFS filesystem. **NOTE** This parameter must be applied only for the new deployments. To restrict **csi-users** per pool and per cluster, you need to create new **csi-users** and new secrets for those **csi-users**.

Example with restricted auth permission:

```
# python3 /etc/ceph/create-external-cluster-resources.py --cephfs-filesystem-name
myfs --rbd-data-pool-name replicapool --cluster-name rookStorage --restricted-auth-
permission true
```

Example of JSON output generated using the python script:

```
[{"name": "rook-ceph-mon-endpoints", "kind": "ConfigMap", "data": {"data":
"xxx.xxx.xxx.xxx:xxxx", "maxMonId": "0", "mapping": "{}"}}, {"name": "rook-ceph-mon",
"kind": "Secret", "data": {"admin-secret": "admin-secret", "fsid": "<fs-id>", "mon-
secret": "mon-secret"}}, {"name": "rook-ceph-operator-creds", "kind": "Secret", "data":
{"userID": "<user-id>", "userKey": "<user-key>"}}, {"name": "rook-csi-rbd-node",
"kind": "Secret", "data": {"userID": "csi-rbd-node", "userKey": "<user-key>"}}, {"name":
"ceph-rbd", "kind": "StorageClass", "data": {"pool": "<pool>"}}, {"name": "monitoring-
endpoint", "kind": "CephCluster", "data": {"MonitoringEndpoint": "xxx.xxx.xxx.xxx",
"MonitoringPort": "xxxx"}}, {"name": "rook-ceph-dashboard-link", "kind": "Secret",
"data": {"userID": "ceph-dashboard-link", "userKey": "<user-key>"}}, {"name": "rook-
csi-rbd-provisioner", "kind": "Secret", "data": {"userID": "csi-rbd-provisioner",
"userKey": "<user-key>"}}, {"name": "rook-csi-cephfs-provisioner", "kind": "Secret",
"data": {"adminID": "csi-cephfs-provisioner", "adminKey": "<admin-key>"}}, {"name":
"rook-csi-cephfs-node", "kind": "Secret", "data": {"adminID": "csi-cephfs-node",
"adminKey": "<admin-key>"}}, {"name": "cephfs", "kind": "StorageClass", "data":
{"fsName": "cephfs", "pool": "cephfs_data"}}, {"name": "ceph-rgw", "kind":
"StorageClass", "data": {"endpoint": "xxx.xxx.xxx.xxx:xxxx", "poolPrefix": "default"}},
{"name": "rgw-admin-ops-user", "kind": "Secret", "data": {"accessKey": "<access-
key>", "secretKey": "<secret-key>"}}]
```

iii. Save the JSON output to a file with **.json** extension

> **NOTE**
>
> For OpenShift Data Foundation to work seamlessly, ensure that the
> parameters (RGW endpoint, CephFS details, RBD pool, and so on) to be
> uploaded using the JSON file remain unchanged on the RHCS external
> cluster after the storage cluster creation.

iv. Run the command when there is a multi-tenant deployment in which the RHCS cluster is
already connected to OpenShift Data Foundation deployment with a lower version.

```
# python3 ceph-external-cluster-details-exporter.py --upgrade
```

d. Click **Browse** to select and upload the JSON file.
The content of the JSON file is populated and displayed in the text box.

e. Click the **Next** button, which is enabled after you upload the **.json** file.

5. In the Review and create page, review the configuration details.
To modify any configuration settings, click **Back** to go back to the previous configuration page.

6. Click **Create StorageSystem**.

Verification steps

- Navigate to **Storage → Data Foundation → Storage Systems** tab and verify that you can view
all storage clusters.

- Verify that all components for the external OpenShift Data Foundation are successfully installed. See Verifying external OpenShift Data Foundation storage deployment for instructions.

CHAPTER 5. VERIFYING EXTERNAL OPENSHIFT DATA FOUNDATION STORAGE CLUSTER DEPLOYMENT

Use this section to verify that the OpenShift Data Foundation deployed as external storage is deployed correctly.

5.1. VERIFYING THE STATE OF THE PODS

1. Click **Workloads → Pods** from the left pane of the OpenShift Web Console.

2. Select **openshift-storage** from the **Project** drop-down list.

   > **NOTE**
   >
   > If the **Show default projects** option is disabled, use the toggle button to list all the default projects.

   For more information on the expected number of pods for each component and how it varies depending on the number of nodes, see Table 5.1, "Pods corresponding to OpenShift Data Foundation components"

3. Verify that the following pods are in running state:

   Table 5.1. Pods corresponding to OpenShift Data Foundation components

Component	Corresponding pods
OpenShift Data Foundation Operator	• **ocs-operator-*** (1 pod on any worker node) • **ocs-metrics-exporter-*** (1 pod on any worker node) > **NOTE** > This pod must be present in **openshift-storage-extended** namespace as well such that there is 1 pod in each **openshift-storage** and **openshift-storage extended** namespace. • **odf-operator-controller-manager-*** (1 pod on any worker node) • **odf-console-*** (1 pod on any worker node) • **csi-addons-controller-manager-*** (1 pod on any worker node)
Rook-ceph Operator	• **rook-ceph-operator-*** (1 pod on any worker node)

Component	Corresponding pods
CSI	**cephfs** **csi-cephfsplugin-*** (1 pod on each worker node) **csi-cephfsplugin-provisioner-*** (2 pods distributed across worker nodes) **rbd** **csi-rbdplugin-*** (1 pod on each worker node) **csi-rbdplugin-provisioner-*** (2 pods distributed across worker nodes)

5.2. VERIFYING THAT THE OPENSHIFT DATA FOUNDATION CLUSTER IS HEALTHY

1. In the OpenShift Web Console, click **Storage → Data Foundation**.

2. In the **Status** card of the **Overview** tab, click **Storage System** and verify that you can now see two storage system links in the pop up that appears.

3. Click each of the storage system links and verify the following:

   a. In the **Status** card of the **Block and File** tab, verify that the *Storage Cluster* has a green tick.

   b. In the **Details** card, verify that the cluster information is displayed.

For more information on the health of the OpenShift Data Foundation cluster using the Block and File dashboard, see Monitoring OpenShift Data Foundation .

5.3. VERIFYING THAT THE MULTICLOUD OBJECT GATEWAY IS HEALTHY

1. In the OpenShift Web Console, click **Storage → Data Foundation**.

2. In the **Status** card of the **Overview** tab, click **Storage System** and then click the storage system link from the pop up that appears.

   a. In the **Status** card of the **Object** tab, verify that both *Object Service* and *Data Resiliency* have a green tick.

   b. In the **Details** card, verify that the Multicloud Object Gateway (MCG) information is displayed.

> **NOTE**
>
> The RADOS Object Gateway is only listed in case RADOS Object Gateway endpoint details are included while deploying OpenShift Data Foundation in external mode.

For more information on the health of OpenShift Data Foundation cluster using the object dashboard, see Monitoring OpenShift Data Foundation .

5.4. VERIFYING THAT THE STORAGE CLASSES ARE CREATED AND LISTED

1. Click **Storage → Storage Classes**from the left pane of the OpenShift Web Console.

2. Verify that the following storage classes are created with the OpenShift Data Foundation cluster creation:

   - **ocs-external-storagecluster-ceph-rbd**

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

   - **ocs-external-storagecluster-cephfs**

   NOTE

   - If an MDS is not deployed in the external cluster, **ocs-external-storagecluster-cephfs** storage class will not be created.

   - If RGW is not deployed in the external cluster, the **ocs-external-storagecluster-ceph-rgw** storage class will not be created.

   For more information regarding MDS and RGW, see Red Hat Ceph Storage documentation

5.5. VERIFYING THAT CEPH CLUSTER IS CONNECTED

Run the following command to verify if the OpenShift Data Foundation cluster is connected to the external Red Hat Ceph Storage cluster.

```
$ oc get cephcluster -n openshift-storage-extended
NAME                          DATADIRHOSTPATH  MONCOUNT  AGE  PHASE     MESSAGE
HEALTH    EXTERNAL  FSID
ocs-external-storagecluster-cephcluster                 51m  Connected  Cluster connected
successfully  HEALTH_OK  true    8f01d842-d4b2-11ee-b43c-0050568fb522
```

5.6. VERIFYING THAT STORAGE CLUSTER IS READY

Run the following command to verify if the storage cluster is ready and the **External** option is set to **true**.

```
$ oc get storagecluster -n openshift-storage-extended
NAME                    AGE  PHASE  EXTERNAL  CREATED AT           VERSION
ocs-external-storagecluster  51m  Ready  true    2024-02-28T10:05:54Z  4.15.0
```

CHAPTER 6. MIGRATING APPLICATION WORKLOADS

You can migrate application workloads from the internal mode storage classes to the external mode storage classes using Migration Toolkit for Containers using the same cluster as source and target.