Red Hat

Red Hat OpenShift Data Foundation 4.12

Deploying OpenShift Data Foundation using bare metal infrastructure

Instructions on deploying OpenShift Data Foundation using local storage on bare metal infrastructure

Last Updated: 2024-03-06

Red Hat OpenShift Data Foundation 4.12 Deploying OpenShift Data Foundation using bare metal infrastructure

Instructions on deploying OpenShift Data Foundation using local storage on bare metal infrastructure

Abstract

Read this document for instructions about how to install Red Hat OpenShift Data Foundation to use local storage on bare metal infrastructure.

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**.

PREFACE

Red Hat OpenShift Data Foundation supports deployment on existing Red Hat OpenShift Container Platform (RHOCP) bare metal clusters in connected or disconnected environments along with out-of-the-box support for proxy environments.

Both internal and external OpenShift Data Foundation clusters are supported on bare metal. See Planning your deployment and Preparing to deploy OpenShift Data Foundation for more information about deployment requirements.

To deploy OpenShift Data Foundation, follow the appropriate deployment process based on your requirement:

- Internal mode
    - Deploy using local storage devices
    - Deploy standalone Multicloud Object Gateway component
- External mode

CHAPTER 1. PREPARING TO DEPLOY OPENSHIFT DATA FOUNDATION

When you deploy OpenShift Data Foundation on OpenShift Container Platform using the local storage devices, you can create internal cluster resources. This approach internally provisions base services so that all the applications can access additional storage classes.

Before you begin the deployment of Red Hat OpenShift Data Foundation using a local storage, ensure that you meet the resource requirements. See Requirements for installing OpenShift Data Foundation using local storage devices.

- Optional: If you want to enable cluster-wide encryption using an external Key Management System (KMS) follow these steps:
  - Ensure that you have a valid Red Hat OpenShift Data Foundation Advanced subscription. To know how subscriptions for OpenShift Data Foundation work, see knowledgebase article on OpenShift Data Foundation subscriptions.
  - When the Token authentication method is selected for encryption, refer to Enabling cluster-wide encryption with the Token authentication using KMS.
  - When the Kubernetes authentication method is selected for encryption then refer to Enabling cluster-wide encryption with KMS using the Kubernetes authentication method .
  - Ensure that you are using signed certificates on your vault servers.

After you have addressed the above, perform the following steps:

1. Install the Local Storage Operator .
2. Install the Red Hat OpenShift Data Foundation Operator .
3. Create the OpenShift Data Foundation cluster on bare metal .

1.1. REQUIREMENTS FOR INSTALLING OPENSHIFT DATA FOUNDATION USING LOCAL STORAGE DEVICES

Node requirements

The cluster must consist of at least three OpenShift Container Platform worker nodes with locally attached-storage devices on each of them.

- Each of the three selected nodes must have at least one raw block device available. OpenShift Data Foundation uses the one or more available raw block devices.
- The devices you use must be empty, the disks must not include Physical Volumes (PVs), Volume Groups (VGs), or Logical Volumes (LVs) remaining on the disk.

For more information, see the *Resource requirements* section in the Planning guide.

Disaster recovery requirements [Technology Preview]

Disaster Recovery features supported by Red Hat OpenShift Data Foundation require all of the following prerequisites to successfully implement a disaster recovery solution:

- A valid Red Hat OpenShift Data Foundation Advanced subscription.

- A valid Red Hat Advanced Cluster Management (RHACM) for Kubernetes subscription.

To know in detail how subscriptions for OpenShift Data Foundation work, see knowledgebase article on OpenShift Data Foundation subscriptions.

For detailed disaster recovery solution requirements, see Configuring OpenShift Data Foundation Disaster Recovery for OpenShift Workloads guide, and *Requirements and recommendations* section of the Install guide in Red Hat Advanced Cluster Management for Kubernetes documentation.

Arbiter stretch cluster requirements [Technology Preview]

In this case, a single cluster is stretched across two zones with a third zone as the location for the arbiter. This is a Technology Preview feature that is currently intended for deployment in the OpenShift Container Platform on-premises and in the same data center. This solution is not recommended for deployments stretching over multiple data centers. Instead, consider Metro-DR as a first option for no data loss DR solution deployed over multiple data centers with low latency networks.

For detailed requirements and instructions, see the Knowledgebase article on *Configuring OpenShift Data Foundation for stretch cluster*.

To know in detail how subscriptions for OpenShift Data Foundation work, see knowledgebase article on OpenShift Data Foundation subscriptions.

NOTE

You cannot enable Flexible scaling and Arbiter both at the same time as they have conflicting scaling logic. With Flexible scaling, you can add one node at a time to your OpenShift Data Foundation cluster. Whereas, in an Arbiter cluster, you need to add at least one node in each of the two data zones.

Compact mode requirements

You can install OpenShift Data Foundation on a three-node OpenShift compact bare-metal cluster, where all the workloads run on three strong master nodes. There are no worker or storage nodes.

To configure OpenShift Container Platform in compact mode, see the *Configuring a three-node cluster* section of the Installing guide in OpenShift Container Platform documentation, and Delivering a Three-node Architecture for Edge Deployments.

Minimum starting node requirements

An OpenShift Data Foundation cluster is deployed with a minimum configuration when the resource requirement for a standard deployment is not met.

For more information, see the *Resource requirements* section in the Planning guide.

CHAPTER 2. DEPLOY OPENSHIFT DATA FOUNDATION USING LOCAL STORAGE DEVICES

You can deploy OpenShift Data Foundation on bare metal infrastructure where OpenShift Container Platform is already installed.

Also, it is possible to deploy only the Multicloud Object Gateway (MCG) component with OpenShift Data Foundation. For more information, see Deploy standalone Multicloud Object Gateway .

Perform the following steps to deploy OpenShift Data Foundation:

1. Install the Local Storage Operator .
2. Install the Red Hat OpenShift Data Foundation Operator .
3. Create an OpenShift Data Foundation cluster on bare metal .

2.1. INSTALLING LOCAL STORAGE OPERATOR

Install the Local Storage Operator from the Operator Hub before creating Red Hat OpenShift Data Foundation clusters on local storage devices.

Procedure

1. Log in to the OpenShift Web Console.
2. Click **Operators → OperatorHub**.
3. Type **local storage** in the **Filter by keyword** box to find the **Local Storage Operator** from the list of operators, and click on it.
4. Set the following options on the **Install Operator** page:
   a. Update channel as either **4.12** or **stable**.
   b. Installation mode as **A specific namespace on the cluster**.
   c. Installed Namespace as **Operator recommended namespace openshift-local-storage**.
   d. Update approval as **Automatic**.
5. Click **Install**.

Verification steps

- Verify that the Local Storage Operator shows a green tick indicating successful installation.

2.2. INSTALLING RED HAT OPENSHIFT DATA FOUNDATION OPERATOR

You can install Red Hat OpenShift Data Foundation Operator using the Red Hat OpenShift Container Platform Operator Hub.

Prerequisites

- Access to an OpenShift Container Platform cluster using an account with **cluster-admin** and operator installation permissions.
- You must have at least three worker nodes in the Red Hat OpenShift Container Platform cluster. Each node should include one disk and requires 3 disks (PVs). However, one PV remains eventually unused by default. This is an expected behavior.
- For additional resource requirements, see the Planning your deployment guide.

IMPORTANT

- When you need to override the cluster-wide default node selector for OpenShift Data Foundation, you can use the following command to specify a blank node selector for the **openshift-storage** namespace (create **openshift-storage** namespace in this case):

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

- Taint a node as **infra** to ensure only Red Hat OpenShift Data Foundation resources are scheduled on that node. This helps you save on subscription costs. For more information, see the *How to use dedicated worker nodes for Red Hat OpenShift Data Foundation* section in the Managing and Allocating Storage Resources guide.

Procedure

1. Log in to the OpenShift Web Console.
2. Click **Operators → OperatorHub**.
3. Scroll or type **OpenShift Data Foundation** into the **Filter by keyword** box to find the **OpenShift Data Foundation** Operator.
4. Click **Install**.
5. Set the following options on the **Install Operator** page:
   a. Update Channel as **stable-4.12**.
   b. Installation Mode as **A specific namespace on the cluster**.
   c. Installed Namespace as **Operator recommended namespace openshift-storage**. If Namespace **openshift-storage** does not exist, it is created during the operator installation.
   d. Select Approval Strategy as **Automatic** or **Manual**.
      If you select **Automatic** updates, then the Operator Lifecycle Manager (OLM) automatically upgrades the running instance of your Operator without any intervention.

      If you select **Manual** updates, then the OLM creates an update request. As a cluster administrator, you must then manually approve that update request to update the Operator to a newer version.
   e. Ensure that the **Enable** option is selected for the **Console plugin**.
   f. Click **Install**.

Verification steps

- After the operator is successfully installed, a pop-up with a message, **Web console update is available** appears on the user interface. Click **Refresh web console** from this pop-up for the console changes to reflect.
- In the Web Console:
  - Navigate to Installed Operators and verify that the **OpenShift Data Foundation** Operator shows a green tick indicating successful installation.
  - Navigate to **Storage** and verify if **Data Foundation** dashboard is available.

2.3. ENABLING CLUSTER-WIDE ENCRYPTION WITH KMS USING THE TOKEN AUTHENTICATION METHOD

You can enable the key value backend path and policy in the vault for token authentication.

Prerequisites

- Administrator access to the vault.
- A valid Red Hat OpenShift Data Foundation Advanced subscription. For more information, see the knowledgebase article on OpenShift Data Foundation subscriptions.
- Carefully, select a unique path name as the backend **path** that follows the naming convention since you cannot change it later.

Procedure

1. Enable the Key/Value (KV) backend path in the vault.
   For vault KV secret engine API, version 1:

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

   For vault KV secret engine API, version 2:

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

2. Create a policy to restrict the users to perform a write or delete operation on the secret:

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

3. Create a token that matches the above policy:

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

2.4. ENABLING CLUSTER-WIDE ENCRYPTION WITH KMS USING THE KUBERNETES AUTHENTICATION METHOD

You can enable the Kubernetes authentication method for cluster-wide encryption using the Key Management System (KMS).

Prerequisites

- Administrator access to Vault.
- A valid Red Hat OpenShift Data Foundation Advanced subscription. For more information, see the knowledgebase article on OpenShift Data Foundation subscriptions.
- The OpenShift Data Foundation operator must be installed from the Operator Hub.
- Select a unique path name as the backend **path** that follows the naming convention carefully. You cannot change this path name later.

Procedure

1. Create a service account:

   ```
   $ oc -n openshift-storage create serviceaccount <serviceaccount_name>
   ```

   where, ***<serviceaccount_name>*** specifies the name of the service account.

   For example:

   ```
   $ oc -n openshift-storage create serviceaccount odf-vault-auth
   ```

2. Create **clusterrolebindings** and **clusterroles**:

   ```
   $ oc -n openshift-storage create clusterrolebinding vault-tokenreview-binding --
   clusterrole=system:auth-delegator --serviceaccount=openshift-
   storage:_<serviceaccount_name>_
   ```

   For example:

   ```
   $ oc -n openshift-storage create clusterrolebinding vault-tokenreview-binding --
   clusterrole=system:auth-delegator --serviceaccount=openshift-storage:odf-vault-auth
   ```

3. Create a secret for the **serviceaccount** token and CA certificate.

   ```
   $ cat <<EOF | oc create -f -
   apiVersion: v1
   kind: Secret
   metadata:
     name: odf-vault-auth-token
     namespace: openshift-storage
     annotations:
       kubernetes.io/service-account.name: <serviceaccount_name>
   type: kubernetes.io/service-account-token
   data: {}
   EOF
   ```

where, ***<serviceaccount_name>*** is the service account created in the earlier step.

4. Get the token and the CA certificate from the secret.

```
$ SA_JWT_TOKEN=$(oc -n openshift-storage get secret odf-vault-auth-token -o jsonpath="
{.data['token']}" | base64 --decode; echo)
$ SA_CA_CRT=$(oc -n openshift-storage get secret odf-vault-auth-token -o jsonpath="
{.data['ca\.crt']}" | base64 --decode; echo)
```

5. Retrieve the OCP cluster endpoint.

```
$ OCP_HOST=$(oc config view --minify --flatten -o jsonpath="{.clusters[0].cluster.server}")
```

6. Fetch the service account issuer:

```
$ oc proxy &
$ proxy_pid=$!
$ issuer="$( curl --silent http://127.0.0.1:8001/.well-known/openid-configuration | jq -r
.issuer)"
$ kill $proxy_pid
```

7. Use the information collected in the previous step to setup the Kubernetes authentication method in Vault:

```
$ vault auth enable kubernetes
```

```
$ vault write auth/kubernetes/config \
        token_reviewer_jwt="$SA_JWT_TOKEN" \
        kubernetes_host="$OCP_HOST" \
        kubernetes_ca_cert="$SA_CA_CRT" \
        issuer="$issuer"
```

IMPORTANT

To configure the Kubernetes authentication method in Vault when the issuer is empty:

```
$ vault write auth/kubernetes/config \
        token_reviewer_jwt="$SA_JWT_TOKEN" \
        kubernetes_host="$OCP_HOST" \
        kubernetes_ca_cert="$SA_CA_CRT"
```

8. Enable the Key/Value (KV) backend path in Vault.
For Vault KV secret engine API, version 1:

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

For Vault KV secret engine API, version 2:

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

9. Create a policy to restrict the users to perform a **write** or **delete** operation on the secret:

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

10. Generate the roles:

```
$ vault write auth/kubernetes/role/odf-rook-ceph-op \
        bound_service_account_names=rook-ceph-system,rook-ceph-osd,noobaa \
        bound_service_account_namespaces=openshift-storage \
        policies=odf \
        ttl=1440h
```

The role **odf-rook-ceph-op** is later used while you configure the KMS connection details during the creation of the storage system.

```
$ vault write auth/kubernetes/role/odf-rook-ceph-osd \
        bound_service_account_names=rook-ceph-osd \
        bound_service_account_namespaces=openshift-storage \
        policies=odf \
        ttl=1440h
```

2.5. CREATING MULTUS NETWORKS [TECHNOLOGY PREVIEW]

OpenShift Container Platform uses the Multus CNI plug-in to allow chaining of CNI plug-ins. You can configure your default pod network during cluster installation. The default network handles all ordinary network traffic for the cluster.

You can define an additional network based on the available CNI plug-ins and attach one or more of these networks to your pods. To attach additional network interfaces to a pod, you must create configurations that define how the interfaces are attached.

You specify each interface by using a NetworkAttachmentDefinition (NAD) custom resource (CR). A CNI configuration inside each of the NetworkAttachmentDefinition defines how that interface is created.

OpenShift Data Foundation uses the CNI plug-in called macvlan. Creating a macvlan-based additional network allows pods on a host to communicate with other hosts and pods on those hosts using a physical network interface. Each pod that is attached to a macvlan-based additional network is provided a unique MAC address.

IMPORTANT

Multus support is a Technology Preview feature that is only supported and has been tested on bare metal and VMWare deployments. Technology Preview features are not supported with Red Hat production service level agreements (SLAs) and might not be functionally complete. Red Hat does not recommend using them in production. These features provide early access to upcoming product features, enabling customers to test functionality and provide feedback during the development process.

For more information, see Technology Preview Features Support Scope .

2.5.1. Creating network attachment definitions

To utilize Multus, an already working cluster with the correct networking configuration is required, see Recommended network configuration and requirements for a Multus configuration . The newly created NetworkAttachmentDefinition (NAD) can be selected during the Storage Cluster installation. This is the reason they must be created before the Storage Cluster.

You can select the newly created **NetworkAttachmentDefinition** (NAD) during the Storage Cluster installation. This is the reason you must create the NAD before you create the Storage Cluster.

As detailed in the Planning Guide, the Multus networks you create depend on the number of available network interfaces you have for OpenShift Data Foundation traffic. It is possible to separate all of the storage traffic onto one of the two interfaces (one interface used for default OpenShift SDN) or to further segregate storage traffic into client storage traffic (public) and storage replication traffic (private or cluster).

The following is an example **NetworkAttachmentDefinition** for all the storage traffic, public and cluster, on the same interface. It requires one additional interface on all schedulable nodes (OpenShift default SDN on separate network interface):

```
apiVersion: "k8s.cni.cncf.io/v1"
kind: NetworkAttachmentDefinition
metadata:
  name: ocs-public-cluster
  namespace: openshift-storage
spec:
  config: '{
   "cniVersion": "0.3.1",
   "type": "macvlan",
   "master": "ens2",
   "mode": "bridge",
   "ipam": {
         "type": "whereabouts",
         "range": "192.168.1.0/24"
   }
  }'
```

NOTE

All network interface names must be the same on all the nodes attached to the Multus network (that is, **ens2** for **ocs-public-cluster**).

The following is an example **NetworkAttachmentDefinition** for storage traffic on separate Multus networks, public, for client storage traffic, and cluster, for replication traffic. It requires two additional interfaces on OpenShift nodes hosting Object Storge Device (OSD) pods and one additional interface on all other schedulable nodes (OpenShift default SDN on separate network interface):

```
apiVersion: "k8s.cni.cncf.io/v1"
kind: NetworkAttachmentDefinition
metadata:
  name: ocs-public
  namespace: openshift-storage
spec:
  config: '{
   "cniVersion": "0.3.1",
   "type": "macvlan",
   "master": "ens2",
   "mode": "bridge",
   "ipam": {
         "type": "whereabouts",
         "range": "192.168.1.0/24"
   }
  }'
```

Example **NetworkAttachmentDefinition**:

```
apiVersion: "k8s.cni.cncf.io/v1"
kind: NetworkAttachmentDefinition
metadata:
  name: ocs-cluster
  namespace: openshift-storage
spec:
  config: '{
   "cniVersion": "0.3.1",
   "type": "macvlan",
   "master": "ens3",
   "mode": "bridge",
   "ipam": {
         "type": "whereabouts",
         "range": "192.168.2.0/24"
   }
  }'
```

NOTE

All network interface names must be the same on all the nodes attached to the Multus networks (that is, **ens2** for **ocs-public**, and **ens3** for **ocs-cluster**).

2.6. CREATING OPENSHIFT DATA FOUNDATION CLUSTER ON BARE METAL

Prerequisites

- Ensure that all the requirements in the Requirements for installing OpenShift Data Foundation using local storage devices section are met.

- If you want to use the Technology Preview feature of multus support, before deployment you must create network attachment definitions (NADs) that is later attached to the cluster. For more information, see Multi network plug-in (Multus) support and Creating network attachment definitions.

Procedure

1. In the OpenShift Web Console, click **Operators → Installed Operators** to view all the installed operators.
   Ensure that the **Project** selected is **openshift-storage**.

2. Click on the **OpenShift Data Foundation** operator, and then click **Create StorageSystem**.

3. In the **Backing storage** page, perform the following:

   a. Select **Full Deployment** for the **Deployment type** option.

   b. Select the **Create a new StorageClass using the local storage devices** option.

   c. Click Next.

   IMPORTANT

   You are prompted to install the Local Storage Operator if it is not already installed. Click **Install**, and follow the procedure as described in Installing Local Storage Operator.

4. In the **Create local volume set** page, provide the following information:

   a. Enter a name for the **LocalVolumeSet** and the **StorageClass**.
      The local volume set name appears as the default value for the storage class name. You can change the name.

   b. Select one of the following:

      - **Disks on all nodes**
        Uses the available disks that match the selected filters on all the nodes.

      - **Disks on selected nodes**
        Uses the available disks that match the selected filters only on the selected nodes.

IMPORTANT

- The flexible scaling feature is enabled only when the storage cluster that you created with three or more nodes are spread across fewer than the minimum requirement of three availability zones.
  For information about flexible scaling, see knowledgebase article on *Scaling OpenShift Data Foundation cluster using YAML when flexible scaling is enabled*.

- Flexible scaling features get enabled at the time of deployment and can not be enabled or disabled later on.

- If the nodes selected do not match the OpenShift Data Foundation cluster requirement of an aggregated 30 CPUs and 72 GiB of RAM, a minimal cluster is deployed.
  For minimum starting node requirements, see the Resource requirements section in the *Planning* guide.

c. From the available list of **Disk Type**, select **SSD/NVMe**.

d. Expand the **Advanced** section and set the following options:

Volume Mode	Block is selected as the default value.
Device Type	Select one or more device type from the dropdown list.
Disk Size	Set a minimum size of 100GB for the device and maximum available size of the device that needs to be included.
Maximum Disks Limit	This indicates the maximum number of Persistent Volumes (PVs) that you can create on a node. If this field is left empty, then PVs are created for all the available disks on the matching nodes.

e. Click **Next**.
   A pop-up to confirm the creation of LocalVolumeSet is displayed.

f. Click **Yes** to continue.

5. In the **Capacity and nodes** page, configure the following:

   a. **Available raw capacity** is populated with the capacity value based on all the attached disks associated with the storage class. This takes some time to show up. The **Selected nodes** list shows the nodes based on the storage class.

   b. Optional: Select the **Taint nodes** checkbox to dedicate the selected nodes for OpenShift Data Foundation.

   c. Click **Next**.

6. Optional: In the **Security and network** page, configure the following based on your requirement:

   a. To enable encryption, select **Enable data encryption for block and file storage**

b. Select one or both of the following **Encryption level**:

- **Cluster-wide encryption**
  Encrypts the entire cluster (block and file).
- **StorageClass encryption**
  Creates encrypted persistent volume (block only) using encryption enabled storage class.

c. Optional: Select the **Connect to an external key management service** checkbox. This is optional for cluster-wide encryption.

i. From the **Key Management Service Provider** drop-down list, either select **Vault** or **Thales CipherTrust Manager (using KMIP)**. If you selected **Vault**, go to the next step. If you selected **Thales CipherTrust Manager (using KMIP)**, go to step iii.

ii. Select an **Authentication Method**.

Using Token authentication method

- Enter a unique **Connection Name**, host **Address** of the Vault server ('https://<hostname or ip>'), **Port** number and **Token**.
- Expand **Advanced Settings** to enter additional settings and certificate details based on your **Vault** configuration:
  - Enter the Key Value secret path in **Backend Path** that is dedicated and unique to OpenShift Data Foundation.
  - Optional: Enter **TLS Server Name** and **Vault Enterprise Namespace**.
  - Upload the respective PEM encoded certificate file to provide the **CA Certificate**, **Client Certificate** and **Client Private Key** .
  - Click **Save** and skip to step iv.

Using Kubernetes authentication method

- Enter a unique Vault **Connection Name**, host **Address** of the Vault server ('https://<hostname or ip>'), **Port** number and **Role** name.
- Expand **Advanced Settings** to enter additional settings and certificate details based on your **Vault** configuration:
  - Enter the Key Value secret path in **Backend Path** that is dedicated and unique to OpenShift Data Foundation.
  - Optional: Enter **TLS Server Name** and **Authentication Path** if applicable.
  - Upload the respective PEM encoded certificate file to provide the **CA Certificate**, **Client Certificate** and **Client Private Key** .
  - Click **Save** and skip to step iv.

iii. To use **Thales CipherTrust Manager (using KMIP)** as the KMS provider, follow the steps below:

A. Enter a unique **Connection Name** for the Key Management service within the project.

B. In the **Address** and **Port** sections, enter the IP of Thales CipherTrust Manager and the port where the KMIP interface is enabled. For example:

- **Address**: 123.34.3.2
- **Port**: 5696

C. Upload the **Client Certificate**, **CA certificate**, and **Client Private Key**.

D. If StorageClass encryption is enabled, enter the Unique Identifier to be used for encryption and decryption generated above.

E. The **TLS Server** field is optional and used when there is no DNS entry for the KMIP endpoint. For example, **kmip_all_<port>.ciphertrustmanager.local**.

iv. Select a **Network**.

d. Select one of the following:

- **Default (SDN)**
  If you are using a single network.
- **Custom (Multus)**
  If you are using multiple network interfaces.

  i. Select a **Public Network Interface** from the dropdown.

  ii. Select a **Cluster Network Interface** from the dropdown.

NOTE

If you are using only one additional network interface, select the single **NetworkAttachementDefinition**, that is,**ocs-public-cluster** for the Public Network Interface and leave the Cluster Network Interface blank.

e. Click **Next**.

7. 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.

8. Click **Create StorageSystem**.

Verification steps

- To verify the final Status of the installed storage cluster:

  a. In the OpenShift Web Console, navigate to **Installed Operators → OpenShift Data Foundation → Storage System**

  b. Click **ocs-storagecluster-storagesystem → Resources**.

  c. Verify that the **Status** of the **StorageCluster** is **Ready** and has a green tick mark next to it.

- To verify if the flexible scaling is enabled on your storage cluster, perform the following steps (for arbiter mode, flexible scaling is disabled):
    1. In the OpenShift Web Console, navigate to **Installed Operators → OpenShift Data Foundation → Storage System**
    2. Click **ocs-storagecluster-storagesystem → Resources → ocs-storagecluster**.
    3. In the YAML tab, search for the keys **flexibleScaling** in the **spec** section and **failureDomain** in the **status** section. If **flexible scaling** is true and **failureDomain** is set to host, flexible scaling feature is enabled:

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

- To verify that all the components for OpenShift Data Foundation are successfully installed, see Verifying your OpenShift Data Foundation installation .
- To verify the multi networking (Multus), see Verifying the Multus networking.

Additional resources

- To expand the capacity of the initial cluster, see the Scaling Storage guide.

2.7. VERIFYING OPENSHIFT DATA FOUNDATION DEPLOYMENT

To verify that OpenShift Data Foundation is deployed correctly:

1. Verify the state of the pods.
2. Verify that the OpenShift Data Foundation cluster is healthy .
3. Verify that the Multicloud Object Gateway is healthy .
4. Verify that the OpenShift Data Foundation specific storage classes exist .
5. Verify the Multus networking.

2.7.1. Verifying the state of the pods

Procedure

1. Click **Workloads → Pods** from 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 2.1, "Pods corresponding to OpenShift Data Foundation cluster".

3. Set filter for Running and Completed pods to verify that the following pods are in **Running** and **Completed** state:

Table 2.1. Pods corresponding to OpenShift Data Foundation cluster

Component	Corresponding pods
OpenShift Data Foundation Operator	• **ocs-operator-*** (1 pod on any storage node) • **ocs-metrics-exporter-*** (1 pod on any storage node) • **odf-operator-controller-manager-*** (1 pod on any storage node) • **odf-console-*** (1 pod on any storage node) • **csi-addons-controller-manager-*** (1 pod on any storage node)
Rook-ceph Operator	**rook-ceph-operator-*** (1 pod on any storage node)
Multicloud Object Gateway	• **noobaa-operator-*** (1 pod on any storage node) • **noobaa-core-*** (1 pod on any storage node) • **noobaa-db-pg-*** (1 pod on any storage node) • **noobaa-endpoint-*** (1 pod on any storage node)
MON	**rook-ceph-mon-*** (3 pods distributed across storage nodes)
MGR	**rook-ceph-mgr-*** (1 pod on any storage node)
MDS	**rook-ceph-mds-ocs-storagecluster-cephfilesystem-*** (2 pods distributed across storage nodes)

Component	Corresponding pods
RGW	**rook-ceph-rgw-ocs-storagecluster-cephobjectstore-*** (1 pod on any storage node)
CSI	• **cephfs** ◦ **csi-cephfsplugin-*** (1 pod on each storage node) ◦ **csi-cephfsplugin-provisioner-*** (2 pods distributed across storage nodes) • **rbd** ◦ **csi-rbdplugin-*** (1 pod on each storage node) ◦ **csi-rbdplugin-provisioner-*** (2 pods distributed across storage nodes)
rook-ceph-crashcollector	**rook-ceph-crashcollector-*** (1 pod on each storage node)
OSD	• **rook-ceph-osd-*** (1 pod for each device) • **rook-ceph-osd-prepare-ocs-deviceset-*** (1 pod for each device)

2.7.2. Verifying the OpenShift Data Foundation cluster is healthy

Procedure

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.
3. In the **Status** card of the **Block and File** tab, verify that *Storage Cluster* has a green tick.
4. 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.

2.7.3. Verifying the Multicloud Object Gateway is healthy

Procedure

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 MCG information is displayed.

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

2.7.4. Verifying that the specific storage classes exist

Procedure

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-storagecluster-ceph-rbd**
    - **ocs-storagecluster-cephfs**
    - **openshift-storage.noobaa.io**
    - **ocs-storagecluster-ceph-rgw**

2.7.5. Verifying the Multus networking

To determine if Multus is working in your cluster, verify the Multus networking.

Procedure

Based on your Network configuration choices, the OpenShift Data Foundation operator will do one of the following:

- If only a single NetworkAttachmentDefinition (for example, **ocs-public-cluster**) was selected for the Public Network Interface, then the traffic between the application pods and the OpenShift Data Foundation cluster will happen on this network. Additionally the cluster will be self configured to also use this network for the replication and rebalancing traffic between OSDs.
- If both NetworkAttachmentDefinitions (for example, **ocs-public** and **ocs-cluster**) were selected for the Public Network Interface and the Cluster Network Interface respectively during the Storage Cluster installation, then client storage traffic will be on the public network and cluster network for the replication and rebalancing traffic between OSDs.

To verify the network configuration is correct, complete the following:

In the OpenShift console, navigate to **Installed Operators → OpenShift Data Foundation → Storage System → ocs-storagecluster-storagesystem → Resources → ocs-storagecluster**.

In the YAML tab, search for **network** in the **spec** section and ensure the configuration is correct for your network interface choices. This example is for separating the client storage traffic from the storage replication traffic.

Sample output:

```
[..]
spec:
  [..]
  network:
    ipFamily: IPv4
    provider: multus
    selectors:
      cluster: openshift-storage/ocs-cluster
      public: openshift-storage/ocs-public
  [..]
```

To verify the network configuration is correct using the command line interface, run the following commands:

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

Sample output:

```
{"ipFamily":"IPv4","provider":"multus","selectors":{"cluster":"openshift-storage/ocs-
cluster","public":"openshift-storage/ocs-public"}}
```

Confirm the OSD pods are using correct network

In the **openshift-storage** namespace use one of the OSD pods to verify the pod has connectivity to the correct networks. This example is for separating the client storage traffic from the storage replication traffic.

NOTE

Only the OSD pods will connect to both Multus public and cluster networks if both are created. All other OCS pods will connect to the Multus public network.

```
$ oc get -n openshift-storage $(oc get pods -n openshift-storage -o name -l app=rook-ceph-osd | grep
'osd-0') -o=jsonpath='{.metadata.annotations.k8s\.v1\.cni\.cncf\.io/network-status}{"\n"}'
```

Sample output:

```
[{
    "name": "openshift-sdn",
    "interface": "eth0",
    "ips": [
        "10.129.2.30"
    ],
    "default": true,
    "dns": {}
},{
    "name": "openshift-storage/ocs-cluster",
    "interface": "net1",
    "ips": [
        "192.168.2.1"
    ],
    "mac": "e2:04:c6:81:52:f1",
    "dns": {}
},{
    "name": "openshift-storage/ocs-public",
    "interface": "net2",
    "ips": [
        "192.168.1.1"
    ],
    "mac": "ee:a0:b6:a4:07:94",
    "dns": {}
}]
```

To confirm the OSD pods are using correct network using the command line interface, run the following command (requires the jq utility):

```
$ oc get -n openshift-storage $(oc get pods -n openshift-storage -o name -l app=rook-ceph-osd | grep
'osd-0') -o=jsonpath='{.metadata.annotations.k8s\.v1\.cni\.cncf\.io/network-status}{"\n"}' | jq -r '.
[].name'
```

Sample output:

```
openshift-sdn
openshift-storage/ocs-cluster
openshift-storage/ocs-public
```

CHAPTER 3. DEPLOY STANDALONE MULTICLOUD OBJECT GATEWAY

Deploying only the Multicloud Object Gateway component with the OpenShift Data Foundation provides the flexibility in deployment and helps to reduce the resource consumption. Use this section to deploy only the standalone Multicloud Object Gateway component, which involves the following steps:

- Installing the Local Storage Operator.
- Installing Red Hat OpenShift Data Foundation Operator
- Creating standalone Multicloud Object Gateway

3.1. INSTALLING LOCAL STORAGE OPERATOR

Install the Local Storage Operator from the Operator Hub before creating Red Hat OpenShift Data Foundation clusters on local storage devices.

Procedure

1. Log in to the OpenShift Web Console.
2. Click **Operators → OperatorHub**.
3. Type **local storage** in the **Filter by keyword** box to find the **Local Storage Operator** from the list of operators, and click on it.
4. Set the following options on the **Install Operator** page:
    a. Update channel as either **4.12** or **stable**.
    b. Installation mode as **A specific namespace on the cluster**.
    c. Installed Namespace as **Operator recommended namespace openshift-local-storage**.
    d. Update approval as **Automatic**.
5. Click **Install**.

Verification steps

- Verify that the Local Storage Operator shows a green tick indicating successful installation.

3.2. INSTALLING RED HAT OPENSHIFT DATA FOUNDATION OPERATOR

You can install Red Hat OpenShift Data Foundation Operator using the Red Hat OpenShift Container Platform Operator Hub.

Prerequisites

- Access to an OpenShift Container Platform cluster using an account with **cluster-admin** and operator installation permissions.

- You must have at least three worker nodes in the Red Hat OpenShift Container Platform cluster. Each node should include one disk and requires 3 disks (PVs). However, one PV remains eventually unused by default. This is an expected behavior.
- For additional resource requirements, see the Planning your deployment guide.

IMPORTANT

- When you need to override the cluster-wide default node selector for OpenShift Data Foundation, you can use the following command to specify a blank node selector for the **openshift-storage** namespace (create **openshift-storage** namespace in this case):

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

- Taint a node as **infra** to ensure only Red Hat OpenShift Data Foundation resources are scheduled on that node. This helps you save on subscription costs. For more information, see the *How to use dedicated worker nodes for Red Hat OpenShift Data Foundation* section in the Managing and Allocating Storage Resources guide.

Procedure

1. Log in to the OpenShift Web Console.
2. Click **Operators → OperatorHub**.
3. Scroll or type **OpenShift Data Foundation** into the **Filter by keyword** box to find the **OpenShift Data Foundation** Operator.
4. Click **Install**.
5. Set the following options on the **Install Operator** page:
   a. Update Channel as **stable-4.12**.
   b. Installation Mode as **A specific namespace on the cluster**.
   c. Installed Namespace as **Operator recommended namespace openshift-storage**. If Namespace **openshift-storage** does not exist, it is created during the operator installation.
   d. Select Approval Strategy as **Automatic** or **Manual**.
      If you select **Automatic** updates, then the Operator Lifecycle Manager (OLM) automatically upgrades the running instance of your Operator without any intervention.

      If you select **Manual** updates, then the OLM creates an update request. As a cluster administrator, you must then manually approve that update request to update the Operator to a newer version.
   e. Ensure that the **Enable** option is selected for the **Console plugin**.
   f. Click **Install**.

Verification steps

- After the operator is successfully installed, a pop-up with a message, **Web console update is available** appears on the user interface. Click **Refresh web console** from this pop-up for the console changes to reflect.
- In the Web Console:
  - Navigate to Installed Operators and verify that the **OpenShift Data Foundation** Operator shows a green tick indicating successful installation.
  - Navigate to **Storage** and verify if **Data Foundation** dashboard is available.

3.3. CREATING A STANDALONE MULTICLOUD OBJECT GATEWAY

You can create only the standalone Multicloud Object Gateway component while deploying OpenShift Data Foundation.

Prerequisites

- Ensure that the OpenShift Data Foundation Operator is installed.

Procedure

1. In the OpenShift Web Console, click **Operators → Installed Operators** to view all the installed operators.
   Ensure that the **Project** selected is **openshift-storage**.
2. Click **OpenShift Data Foundation** operator and then click **Create StorageSystem**.
3. In the **Backing storage** page, select the following:
   a. Select **Multicloud Object Gateway** for **Deployment type**.
   b. Select the **Create a new StorageClass using the local storage devices**option.
   c. Click **Next**.

   NOTE

   You are prompted to install the Local Storage Operator if it is not already installed. Click **Install**, and follow the procedure as described in Installing Local Storage Operator.
4. In the **Create local volume set** page, provide the following information:
   a. Enter a name for the **LocalVolumeSet** and the **StorageClass**.
      By default, the local volume set name appears for the storage class name. You can change the name.
   b. Choose one of the following:
      - **Disks on all nodes**
        Uses the available disks that match the selected filters on all the nodes.
      - **Disks on selected nodes**
        Uses the available disks that match the selected filters only on the selected nodes.

c. From the available list of **Disk Type**, select **SSD/NVMe**.

d. Expand the **Advanced** section and set the following options:

Volume Mode	Filesystem is selected by default. Always ensure that Filesystem is selected for **Volume Mode**.
Device Type	Select one or more device type from the dropdown list.
Disk Size	Set a minimum size of 100GB for the device and maximum available size of the device that needs to be included.
Maximum Disks Limit	This indicates the maximum number of PVs that can be created on a node. If this field is left empty, then PVs are created for all the available disks on the matching nodes.

e. Click **Next**.
A pop-up to confirm the creation of LocalVolumeSet is displayed.

f. Click **Yes** to continue.

5. In the **Capacity and nodes** page, configure the following:

   a. **Available raw capacity** is populated with the capacity value based on all the attached disks associated with the storage class. This takes some time to show up. The **Selected nodes** list shows the nodes based on the storage class.

   b. Click **Next**.

6. Optional: Select the **Connect to an external key management service** checkbox. This is optional for cluster-wide encryption.

   a. From the **Key Management Service Provider** drop-down list, either select **Vault** or **Thales CipherTrust Manager (using KMIP)**. If you selected **Vault**, go to the next step. If you selected **Thales CipherTrust Manager (using KMIP)**, go to step iii.

   b. Select an **Authentication Method**.

   Using Token authentication method

   - Enter a unique **Connection Name**, host **Address** of the Vault server ('https://<hostname or ip>'), **Port** number and **Token**.
   - Expand **Advanced Settings** to enter additional settings and certificate details based on your **Vault** configuration:
     - Enter the Key Value secret path in **Backend Path** that is dedicated and unique to OpenShift Data Foundation.
     - Optional: Enter **TLS Server Name** and **Vault Enterprise Namespace**.
     - Upload the respective PEM encoded certificate file to provide the **CA Certificate**, **Client Certificate** and **Client Private Key** .
     - Click **Save** and skip to step iv.

Using Kubernetes authentication method

- Enter a unique Vault **Connection Name**, host **Address** of the Vault server ('https://<hostname or ip>'), **Port** number and **Role** name.
- Expand **Advanced Settings** to enter additional settings and certificate details based on your **Vault** configuration:
    - Enter the Key Value secret path in **Backend Path** that is dedicated and unique to OpenShift Data Foundation.
    - Optional: Enter **TLS Server Name** and **Authentication Path** if applicable.
    - Upload the respective PEM encoded certificate file to provide the **CA Certificate**, **Client Certificate** and **Client Private Key** .
    - Click **Save** and skip to step iv.

c. To use **Thales CipherTrust Manager (using KMIP)** as the KMS provider, follow the steps below:

  i. Enter a unique **Connection Name** for the Key Management service within the project.

  ii. In the **Address** and **Port** sections, enter the IP of Thales CipherTrust Manager and the port where the KMIP interface is enabled. For example:
   - **Address**: 123.34.3.2
   - **Port**: 5696

  iii. Upload the **Client Certificate**, **CA certificate**, and **Client Private Key**.

  iv. If StorageClass encryption is enabled, enter the Unique Identifier to be used for encryption and decryption generated above.

  v. The **TLS Server** field is optional and used when there is no DNS entry for the KMIP endpoint. For example, **kmip_all_<port>.ciphertrustmanager.local**.

d. Select a **Network**.

e. Click **Next**.

7. In the **Review and create** page, review the configuration details:
   To modify any configuration settings, click **Back**.
8. Click **Create StorageSystem**.

Verification steps

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 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 MCG information is displayed.

Verifying the state of the pods

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

2. Select **openshift-storage** from the **Project** drop-down list and verify that the following pods are in **Running** state.

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

Component	Corresponding pods
OpenShift Data Foundation Operator	● **ocs-operator-*** (1 pod on any storage node) ● **ocs-metrics-exporter-*** (1 pod on any storage node) ● **odf-operator-controller-manager-*** (1 pod on any storage node) ● **odf-console-*** (1 pod on any storage node) ● **csi-addons-controller-manager-*** (1 pod on any storage node)
Rook-ceph Operator	**rook-ceph-operator-*** (1 pod on any storage node)
Multicloud Object Gateway	● **noobaa-operator-*** (1 pod on any storage node) ● **noobaa-core-*** (1 pod on any storage node) ● **noobaa-db-pg-*** (1 pod on any storage node) ● **noobaa-endpoint-*** (1 pod on any storage node)

CHAPTER 4. UNINSTALLING OPENSHIFT DATA FOUNDATION

4.1. UNINSTALLING OPENSHIFT DATA FOUNDATION IN INTERNAL MODE

To uninstall OpenShift Data Foundation in Internal mode, refer to the knowledge base article on Uninstalling OpenShift Data Foundation.