Red Hat OpenShift Data Foundation 4.9

Metro-DR ストレッチクラスター用の OpenShift Data Foundation の設定

災害復旧機能を備えたストレージインフラストラクチャーを提供するために、2 つの異なる地理的ロケーション間で OpenShift Data Foundation を設定する方法

Last Updated: 2023-02-03

Red Hat OpenShift Data Foundation 4.9 Metro-DR ストレッチクラスター用の OpenShift Data Foundation の設定

災害復旧機能を備えたストレージインフラストラクチャーを提供するために、2 つの異なる地理的ロケーション間で OpenShift Data Foundation を設定する方法

Enter your first name here. Enter your surname here.
Enter your organisation's name here. Enter your organisational division here.
Enter your email address here.

法律上の通知

概要

このソリューションガイドの目的は、OpenShift Data Foundation を Kubernetes ゾーントポロジーラベルと共にデプロイするのに必要な手順およびコマンドの詳細を提供し、可用性の高いストレージインフラストラクチャーを実現します。 Configuring OpenShift Data Foundation for Metro-DR is a technology preview feature. 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.

目次

多様性を受け入れるオープンソースの強化

Red Hat では、コード、ドキュメント、Web プロパティーにおける配慮に欠ける用語の置き換えに取り組んでいます。まずは、マスター (master)、スレーブ (slave)、ブラックリスト (blacklist)、ホワイトリスト (whitelist) の 4 つの用語の置き換えから始めます。この取り組みは膨大な作業を要するため、今後の複数のリリースで段階的に用語の置き換えを実施して参ります。詳細は、Red Hat CTO である Chris Wright のメッセージ をご覧ください。

RED HAT ドキュメントへのフィードバック (英語のみ)

弊社のドキュメントについてのご意見をお聞かせください。ドキュメントの改善点があれば、ぜひお知らせください。フィードバックをお寄せいただくには、以下をご確認ください。

- 特定の部分についての簡単なコメントをお寄せいただく場合は、以下をご確認ください。
  1. ドキュメントの表示が **Multi-page HTML** 形式になっていていることを確認してください。ドキュメントの右上隅に **Feedback** ボタンがあることを確認してください。
  2. マウスカーソルを使用して、コメントを追加するテキストの部分を強調表示します。
  3. 強調表示されたテキストの下に表示される **Add Feedback** ポップアップをクリックします。
  4. 表示される指示に従ってください。
- より詳細なフィードバックをお寄せいただく場合は、Bugzilla のチケットを作成してください。
  1. Bugzilla の Web サイトに移動します。
  2. **Component** セクションで、**documentation** を選択します。
  3. **Description** フィールドに、ドキュメントの改善に向けたご提案を記入してください。ドキュメントの該当部分へのリンクも追加してください。
  4. **Submit Bug** をクリックします。

第1章 METRO-DR ストレッチクラスターの概要

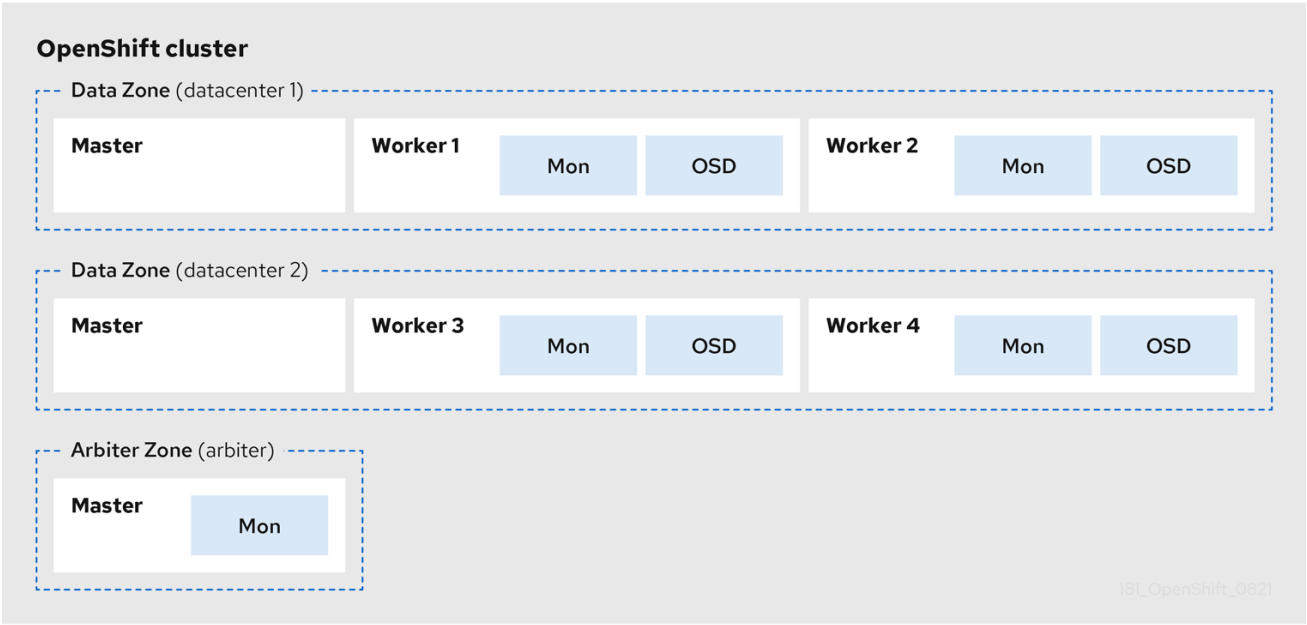
Red Hat OpenShift Data Foundation デプロイメントは、災害復旧機能を備えたストレージインフラストラクチャーを提供するために、2 つの異なる地理的ロケーション間で展開できます。2 つの拠点のうちどちらかが部分的または完全に利用できないといった災害に直面した場合、OpenShift Data Foundation のデプロイメント上にデプロイされた OpenShift Container Storage が存続できるようにしなければなりません。このソリューションは、インフラストラクチャーのサーバー間に特定のレイテンシー要件があるメトロポリタンに広がるデータセンターでのみ利用できます。

注記

現在、レイテンシーが、異なる場所にある OpenShift Container Platform ノード間の 4 ミリ秒のラウンドトリップタイム (RTT) を超えないストレッチクラスターを使用して Metro-DR ソリューションをデプロイできます。より高いレイテンシーでデプロイする予定がある場合は、Red Hat カスタマーサポート にお問い合わせください。

以下の図は、Metro-DR ストレッチクラスターの最も簡単なデプロイメントを示しています。

OpenShift ノードおよび OpenShift Data Foundation デーモン

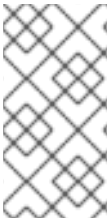


この図では、Arbiter ゾーンにデプロイされた OpenShift Data Foundation モニターの Pod にはマスターノード向けの耐性が組み込まれています。この図では、高可用性 OpenShift Container Platform コントロールプレーンに必要な各データゾーンのマスターノードを示しています。また、いずれかのゾーンの OpenShift Container Platform ノードに、他の 2 つのゾーンの OpenShift Container Platform ノードとのネットワーク接続があることが重要です。

第2章 災害復旧を有効にしてストレージクラスターをデプロイする準備

2.1. METRO-DR を有効にする要件

- 3 つの異なるゾーンに 3 つ以上の OpenShift Container Platform マスターノードがあることを確認してください。3 つのゾーンのそれぞれに 1 つのマスターノード。
- また、4 つ以上の OpenShift Container Platform ワーカーノードが 2 つの Data Zone に均等に分散されていることを確認します。
- ベアメタル上のストレッチクラスターの場合、SSD ドライブを OpenShift Container Platform マスターノードのルートドライブとして使用します。
- 各ノードのゾーンラベルで事前にラベル付けされていることを確認します。詳細は、Applying topology zone labels to OpenShift Container Platform node セクションを参照してください。
- Metro-DR ソリューションは、最大 4 ミリ秒のラウンドトリップタイム (RTT) を使用してレイテンシーがゾーン間で 2 ミリ秒を超えないデプロイメント向けに設計されています。より高いレイテンシーでデプロイする予定がある場合は、Red Hat カスタマーサポート にお問い合わせください。



注記

柔軟なスケーリングおよび Arbiter はどちらもスケーリングロジックの競合がある場合も同時に有効にすることはできません。Flexible scaling を使用すると、1 度に 1 つのノードを OpenShift Data Foundation クラスターに追加することができます。Arbiter クラスターでは、2 つのデータゾーンごとに 1 つ以上のノードを追加する必要があります。

2.2. トポロジーゾーンラベルの OPENSHIFT CONTAINER PLATFORM ノードへの適用

サイトの停止時に、arbiter 関数を持つゾーンは arbiter ラベルを使用します。これらのラベルは任意となるため、3 つの場所で一意にする必要があります。

たとえば、以下のようにノードにラベルを付けることができます。

```
topology.kubernetes.io/zone=arbiter for Master0

topology.kubernetes.io/zone=datacenter1 for Master1, Worker1, Worker2

topology.kubernetes.io/zone=datacenter2 for Master2, Worker3, Worker4
```

- ラベルをノードに適用するには、以下を実行します。

```
$ oc label node <NODENAME> topology.kubernetes.io/zone=<LABEL>
```

<NODENAME>
ノードの名前です。

<LABEL>
トポロジーゾーンラベルです。

- 3 つのゾーンのサンプルラベルを使用してラベルを検証するには、以下を実行します。

```
$ oc get nodes -l topology.kubernetes.io/zone=<LABEL> -o name
```

<LABEL>
トポロジーゾーンラベルです。
または、1 つのコマンドを実行して、そのゾーンを持つすべてのノードを表示できます。

```
$ oc get nodes -L topology.kubernetes.io/zone
```

Metro-DR ストレッチクラスタートポロジーのゾーンラベルが適切な OpenShift Container Platform ノードに適用され、3 つのロケーションが定義されました。

次のステップ

- OpenShift Container Platform OperatorHub からのストレージ Operator のインストール

2.3. ローカルストレージ OPERATOR のインストール

ローカルストレージデバイスに Red Hat OpenShift Data Foundation クラスターを作成する前に、Operator Hub からローカルストレージ Operator をインストールします。

手順

1. OpenShift Web コンソールにログインします。
2. **Operators → OperatorHub** をクリックします。
3. **Filter by keyword** ボックスに **local storage** を入力し、Operator の一覧から **Local Storage Operator** を見つけ、これをクリックします。
4. **Install Operator** ページで、以下のオプションを設定します。
   a. チャンネルを **4.9** または **stable** のいずれかにして更新します。
   b. インストールモードに **A specific namespace on the cluster**を選択します。
   c. Installed Namespace に **Operator recommended namespace openshift-local-storage** を選択します。
   d. 承認を **Automatic** として更新します。
5. **Install** をクリックします。

検証手順

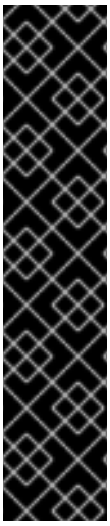
- Local Storage Operator に、インストールが正常に実行されたことを示す緑色のチェックマークが表示されていることを確認します。

2.4. RED HAT OPENSHIFT DATA FOUNDATION OPERATOR のインストール

Red Hat OpenShift Data Foundation Operator は、Red Hat OpenShift Container Platform Operator Hub を使用してインストールできます。

前提条件

- cluster-admin および Operator インストールのパーミッションを持つアカウントを使用して OpenShift Container Platform クラスターにアクセスできること。
- Red Hat OpenShift Container Platform クラスター内の 2 つのデータセンターに均等に分散された 4 ワーカーノードが必要になります。
- その他のリソース要件については、デプロイメントのプランニング を参照してください。



重要

- OpenShift Data Foundation のクラスター全体でのデフォルトノードセレクターを上書きする必要がある場合は、コマンドラインインターフェイスで以下のコマンドを使用し、**openshift-storage** namespace の空のノードセレクターを指定できます (この場合、openshift-storage namespace を作成します)。

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

- ノードに Red Hat OpenShift Data Foundation リソースのみがスケジュールされるように **infra** のテイントを設定します。これにより、サブスクリプションコストを節約できます。詳細は、**ストレージリソースの管理および割り当て**ガイドのHow to use dedicated worker nodes for Red Hat OpenShift Data Foundation の章を参照してください。

手順

1. OpenShift Web コンソールにログインします。
2. **Operators → OperatorHub** をクリックします。
3. スクロールするか、または **OpenShift Data Foundation** を **Filter by keyword** ボックスに入力し、OpenShift **Data Foundation** Operator を検索します。
4. **Install** をクリックします。
5. **Install Operator** ページで、以下のオプションを設定します。
   a. Channel を **stable-4.9** として更新します。
   b. Installation Mode オプションに **A specific namespace on the cluster**を選択します。
   c. Installed Namespace に **Operator recommended namespace openshift-storage** を選択します。namespace **openshift-storage** が存在しない場合、これは Operator のインストール時に作成されます。
   d. **承認ストラテジー** を **Automatic** または **Manual** として選択している。
   Automatic (自動) 更新を選択した場合、Operator Lifecycle Manager (OLM) は介入なしに、Operator の実行中のインスタンスを自動的にアップグレードします。

   Manual 更新を選択した場合、OLM は更新要求を作成します。クラスター管理者は、Operator を新しいバージョンに更新できるように更新要求を手動で承認する必要があります。

6. **Console プラグイン** に **Enable** オプションが選択されていることを確認します。
7. **Install** をクリックします。

検証手順

OpenShift Data Foundation Operator に、インストールが正常に実行されたことを示す緑色のチェックマークが表示されていることを確認します。

次のステップ

- OpenShift Data Foundation クラスターの作成

第3章 OPENSHIFT DATA FOUNDATION クラスターの作成

前提条件

- Preparing to deploy storage cluster with disaster recovery enabled セクションのすべての要件を満たしていることを確認してください。

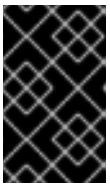
手順

1. OpenShift Web コンソールで、**Operators → Installed Operators**をクリックし、インストールされた Operator を表示します。
   選択された **Project** が **openshift-storage** であることを確認します。

2. **OpenShift Data Foundation** Operator をクリックした後、**Create StorageSystem** をクリックします。

3. Backing storage ページで、**Create a new StorageClass using the local storage devices**オプションを選択します。

4. **Next** をクリックします。



重要

インストールされていない場合に、ローカルストレージ Operator をインストールすることを求めるプロンプトが出されます。**Install** をクリックし、ローカルストレージ Operator のインストールで説明されているように手順に従います。

5. **Create local volume set**ページで、以下の情報を提供します。

   a. **LocalVolumeSet** および **StorageClass** の名前を入力します。
   デフォルトで、ローカルボリュームセット名がストレージクラス名について表示されます。名前を変更できます。

   b. 以下のいずれかを選択します。

   - **Disks on all nodes**
     すべてのノードにある選択したフィルターに一致する利用可能なディスクを使用します。

   - **Disks on selected nodes**
     選択したノードにある選択したフィルターにのみ一致する利用可能なディスクを使用します。

重要

選択したノードが集約された 30 CPU および 72 GiB の RAM の OpenShift Data Foundation クラスターの要件と一致しない場合は、最小クラスターがデプロイされます。

ノードの最小要件については、**プランニング**ガイドのリソース要件セクションを参照してください。

   c. **SSD** または **NVMe** を選択して、サポートされる設定を構築します。サポート対象外のテストインストールに **HDD** を選択できます。

d. **Advanced** セクションを拡張し、以下のオプションを設定します。

ボリュームモード	デフォルトではブロックが選択されます。
デバイスタイプ	ドロップダウンリストから 1 つ以上のデバイスタイプを選択します。
ディスクサイズ	デバイスの最小サイズ 100GB と、含める必要のあるデバイスの最大サイズを設定します。
ディスクの最大数の制限	これは、ノードで作成できる PV の最大数を示します。このフィールドが空のままの場合、PV は一致するノードで利用可能なすべてのディスクに作成されます。

e. **Next** をクリックします。
LocalVolumeSet の作成を確認するポップアップが表示されます。

f. **Yes** をクリックして続行します。

6. **Capacity and nodes** ページで、以下を設定します。

a. ストレッチクラスターを使用する必要がある場合は、**Enable arbiter** チェックボックスを選択します。このオプションは、arbiter のすべての前提条件が満たされ、選択されたノードが設定される場合にのみ利用できます。詳細は、Preparing to deploy storage cluster with disaster recovery enabled [Technology Preview] の Arbiter ストレッチクラスターの要件を参照してください。
ドロップダウンリストから **arbiter ゾーン** を選択します。

b. **Available raw capacity** には、ストレージクラスに関連付けられた割り当てられたすべてのディスクに基づいて容量の値が設定されます。これには少し時間がかかります。
Selected nodes 一覧には、ストレージクラスに基づくノードが表示されます。

c. **Next** をクリックします。

7. オプション: **Security and network** ページで、要件に応じて以下を設定します。

a. **Enable encryption** チェックボックスを選択して、ブロックおよびファイルストレージを暗号化します。

b. 以下の **Encryption level** のいずれかまたは両方を選択します。

- **クラスター全体の暗号化**
クラスター全体を暗号化します (ブロックおよびファイル)。

- **StorageClass の暗号化**
暗号化対応のストレージクラスを使用して、暗号化された永続ボリューム (ブロックのみ) を作成します。

c. **Connect to an external key management service**チェックボックスを選択します。これはクラスター全体の暗号化の場合はオプションになります。

i. **Key Management Service Provider**はデフォルトで **Vault** に設定されます。

ii. Vault **Service Name**、Vault サーバーのホスト **Address**('https://**<hostname or ip>**')、**Port** 番号および **Token** を入力します。

d. **Advanced Settings** を展開して、Vault 設定に基づいて追加の設定および証明書の詳細を入力します。

i. OpenShift Data Foundation 専用で固有のキーバリューシークレットパスを **Backend Path** に入力します。

ii. オプション: **TLS サーバー名** と **Vault Enterprise ネームスペース** を入力します。

iii. それぞれの PEM でエンコードされた証明書ファイルをアップロードし、**CA 証明書**、**クライアント証明書**、および **クライアントの秘密鍵** を提供します。

e. **Save** をクリックします。

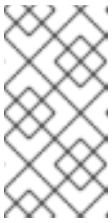
f. 以下のいずれかを選択します。

- **Default (SDN)**
  単一のネットワークを使用している場合。

- **Custom (Multus)**
  複数のネットワークインターフェイスを使用している場合。

i. ドロップダウンメニューから **Public Network Interface** を選択します。

ii. ドロップダウンメニューから **Cluster Network Interface** を選択します。



注記

追加のネットワークインターフェイスを 1 つだけ使用している場合は、単一の**NetworkAttachementDefinition**(Public Network Interface には**ocs-public-cluster**) を選択し、Cluster Network Interface は空白のままにします。

g. **Next** をクリックします。

8. **Review and create** ページで、設定の詳細を確認します。
   設定を変更するには、**Back** をクリックして以前の設定ページに戻ります。

9. **Create StorageSystem** をクリックします。

10. Key Management System (KMS) でのクラスター全体の暗号化の場合、Vault Key/Value (KV) シークレットエンジン API(バージョン 2) を使用している場合は、**configmap** を編集する必要があります。

a. OpenShift Web コンソールで **Workloads → ConfigMaps** に移動します。

b. KMS 接続の詳細を表示するには、ocs **-kms-connection-details** をクリックします。

c. configmap を編集します。

i. **Action menu(⋮)→ Edit ConfigMap** をクリックします。

ii. **VAULT_BACKEND** パラメーターを **v2** に設定します。

```
kind: ConfigMap
```

```
apiVersion: v1
metadata:
  name: ocs-kms-connection-details
[...]
data:
  KMS_PROVIDER: vault
  KMS_SERVICE_NAME: vault
[...]
  VAULT_BACKEND: v2
[...]
```

iii. **Save** をクリックします。

検証手順

- インストールされたストレージクラスターの最終ステータスを確認するには、以下を実行します。

  a. OpenShift Web コンソールで、**Installed Operators → OpenShift Data Foundation → Storage System → ocs-storagecluster-storagesystem → Resources** の順に移動します。

  b. **StorageCluster** の **Status** が **Ready** になっており、それの横に緑色のチェックマークが表示されていることを確認します。

- デプロイメントの arbiter モードの場合:

  a. OpenShift Web コンソールで、**Installed Operators → OpenShift Data Foundation → Storage System → ocs-storagecluster-storagesystem → Resources → ocs-storagecluster** の順に移動します。

  b. YAML タブで、**spec** セクションで **arbiter** キーを検索し、**enable** が **true** に設定されていることを確認します。

```
spec:
   arbiter:
     enable: true
   [..]
   nodeTopologies:
     arbiterLocation: arbiter #arbiter zone
   storageDeviceSets:
   - config: {}
     count: 1
       [..]
     replica: 4
status:
   conditions:
   [..]
   failureDomain: zone
```

- OpenShift Data Foundation のすべてのコンポーネントが正常にインストールされていることを確認するには、OpenShift Data Foundation インストールの確認 を参照してください。

第4章 OPENSHIFT DATA FOUNDATION デプロイメントの確認

OpenShift Data Foundation が正常にデプロイされていることを確認するには、以下を実行します。

- Pod の状態を確認します。
- OpenShift Data Foundation クラスターが正常であることを確認します。
- Multicloud Object Gateway が正常であることを確認
- OpenShift Data Foundation 固有のストレージクラスが存在することを確認します。

4.1. POD の状態の確認

手順

1. OpenShift Web コンソールから **Workloads → Pods** をクリックします。
2. **Project** ドロップダウンリストから **openshift-storage** を選択します。

注記

Show default projects オプションが無効になっている場合は、切り替えボタンを使用して、すべてのデフォルトプロジェクトを一覧表示します。

各コンポーネントについて予想される Pod 数や、これがノード数によってどのように異なるかの詳細は、表4.1「OpenShift Data Foundation クラスターに対応する Pod」 を参照してください。

3. **Running** タブおよび **Completed** タブをクリックして、以下の Pod が **Running** 状態および **Completed** 状態にあることを確認します。

表4.1 OpenShift Data Foundation クラスターに対応する Pod

コンポーネント	対応する Pod
OpenShift Data Foundation Operator	• **ocs-operator-*** (任意のワーカーノードに 1 Pod) • **ocs-metrics-exporter-***(任意のワーカーノードに 1 Pod) • **odf-operator-controller-manager-***(任意のワーカーノードに 1 Pod) • **odf-console-***(任意のワーカーノードに 1 Pod)
Rook-ceph Operator	**rook-ceph-operator-*** (任意のワーカーノードに 1 Pod)

コンポーネント	対応する Pod
Multicloud Object Gateway	• **noobaa-operator-*** (任意のワーカーノードに 1 Pod) • **noobaa-core-*** (任意のストレージノードに 1 Pod) • **nooba-db-*** (任意のストレージノードに 1 Pod) • **noobaa-endpoint-*** (任意のストレージノードに 1 Pod)
MON	**rook-ceph-mon-*** (5 Pod は data-center ゾーンごとに 2 つ、arbiter ゾーンに 1 つと、3 つのゾーンに分散されます)。
MGR	**rook-ceph-mgr-*** (任意のストレージノード上の 2 つの Pod)
MDS	**rook-ceph-mds-ocs-storagecluster-cephfilesystem-*** (2 つの Pod が 2 つのデータセンターゾーンに分散されている)
RGW	**rook-ceph-rgw-ocs-storagecluster-cephobjectstore-*** (2 つの Pod が 2 つのデータセンターゾーンに分散されている)

CSI	• **cephfs**
	  ◦ **csi-cephfsplugin-*** (各ワーカーノードに 1 Pod)
	  ◦ **csi-cephfsplugin-provisioner-*** (ワーカーノードに分散する 2 Pod)
	• **rbd**
	  ◦ **csi-rbdplugin-*** (各ワーカーノードに 1 Pod)
	  ◦ **csi-rbdplugin-provisioner-*** (ストレージノードに分散する 2 Pod)

rook-ceph-crashcollector	**rook-ceph-crashcollector-*** (各ストレージノードに 1 Pod、arbiter ゾーンの 1 Pod)

コンポーネント	対応する Pod
OSD	• **rook-ceph-osd-*** (各デバイス用に 1 Pod) • **rook-ceph-osd-prepare-ocs-deviceset-*** (各デバイス用に 1 Pod)

4.2. OPENSHIFT DATA FOUNDATION クラスターの正常性の確認

手順

1. OpenShift Web コンソールで、**Storage → OpenShift Data Foundation** をクリックします。
2. **Overview** タブの **Status** カードで **Storage System** をクリックし、表示されたポップアップからストレージシステムリンクをクリックします。
3. **Block and File** タブの **Status** カードで、**Storage Cluster** に緑色のチェックマークが表示されていることを確認します。
4. **Details** カードで、クラスター情報が表示されていることを確認します。

ブロックおよびファイルダッシュボードを使用した OpenShift Data Foundation クラスターの正常性については、Monitoring OpenShift Data Foundation を参照してください。

4.3. MULTICLOUD OBJECT GATEWAY が正常であることの確認

手順

1. OpenShift Web コンソールで、**Storage → OpenShift Data Foundation** をクリックします。
2. **Overview** タブの **Status** カードで **Storage System** をクリックし、表示されたポップアップからストレージシステムリンクをクリックします。
   a. **Object** タブの **Status** カードで、**Object Service** と **Data Resiliency** の両方に緑色のチェックマークが表示されていることを確認します。
   b. **Details** カードで、MCG 情報が表示されることを確認します。

ブロックおよびファイルダッシュボードを使用した OpenShift Data Foundation クラスターの正常性については、OpenShift Data Foundation の監視 を参照してください。

4.4. OPENSHIFT DATA FOUNDATION 固有のストレージクラスが存在することの確認

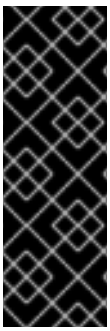
手順

1. OpenShift Web コンソールの左側のペインから **Storage → Storage Classes**をクリックします。
2. 以下のストレージクラスが OpenShift Data Foundation クラスターの作成時に作成されることを確認します。

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

第5章 ゾーン対応サンプルアプリケーションのインストール

ゾーン対応サンプルアプリケーションをデプロイし、OpenShift Data Foundation、Metro-DR 設定が正しく設定されているかどうかを検証します。



重要

データゾーン間のレイテンシーがあると、ノードやゾーン間のレイテンシーが低い (たとえば、すべてのノードが同じ場所にある) OpenShift クラスターと比較して、パフォーマンスの低下が予想されます。どの程度パフォーマンスが低下するかは、ゾーン間のレイテンシーや、ストレージを使用するアプリケーションの動作 (書き込みトラフィックが多いなど) によって異なります。必要なサービスレベルに対して十分なアプリケーションのパフォーマンスを確保するために、必ず Metro DR クラスター設定で重要なアプリケーションをテストしてください。

5.1. ゾーン認識サンプルアプリケーションのインストール

ReadWriteMany (RWX) Persistent Volume Claim (PVC) は、**ocs-storagecluster-cephfs** ストレージクラスを使用して作成されます。複数の Pod は、新規に作成された RWX PVC を同時に使用します。使用されるアプリケーションは File Uploader と呼ばれます。

アプリケーションがトポロジーゾーン全体に分散されるかについてのデモンストレーションにより、サイトが停止した場合にアプリケーションが引き続き利用可能となります。

注記

このアプリケーションはファイルを保存するために同じ RWX ボリュームを共有するため、このデモンストレーションが可能です。Red Hat OpenShift Data Foundation は、ゾーン認識および高可用性を備えた Metro DR ストレッチクラスターとして設定されているため、これは永続的なデータアクセスにも有効です。

1. 新しいプロジェクトを作成する。

```
$ oc new-project my-shared-storage
```

2. file-uploader という名前の PHP アプリケーションのサンプルをデプロイします。

```
$ oc new-app openshift/php:7.3-ubi8~https://github.com/christianh814/openshift-php-upload-demo --name=file-uploader
```

出力例:

```
Found image 4f2dcc0 (9 days old) in image stream "openshift/php" under tag "7.2-ubi8" for
"openshift/php:7.2-
ubi8"

Apache 2.4 with PHP 7.2
-----------------------
PHP 7.2 available as container is a base platform for building and running various PHP 7.2
applications and frameworks. PHP is an HTML-embedded scripting language. PHP attempts
to make it easy for developers to write dynamically generated web pages. PHP also offers
built-in database integration for several commercial and non-commercial database
management systems, so writing a database-enabled webpage with PHP is fairly simple.
```

```
The most common
use of PHP coding is probably as a replacement for CGI scripts.

Tags: builder, php, php72, php-72

* A source build using source code from https://github.com/christianh814/openshift-php-
upload-demo will be cr
eated
* The resulting image will be pushed to image stream tag "file-uploader:latest"
* Use 'oc start-build' to trigger a new build

--> Creating resources ...
    imagestream.image.openshift.io "file-uploader" created
    buildconfig.build.openshift.io "file-uploader" created
    deployment.apps "file-uploader" created
    service "file-uploader" created
--> Success
    Build scheduled, use 'oc logs -f buildconfig/file-uploader' to track its progress.

    Application is not exposed. You can expose services to the outside world by executing one
or more of the commands below:
     'oc expose service/file-uploader'

    Run 'oc status' to view your app.
```

3. ビルドログを表示し、アプリケーションがデプロイされるまで待機します。

```
$ oc logs -f bc/file-uploader -n my-shared-storage
```

出力例:

```
Cloning "https://github.com/christianh814/openshift-php-upload-demo" ...

    [...]
    Generating dockerfile with builder image image-registry.openshift-image-regis
try.svc:5000/openshift/php@sha256:d97466f33999951739a76bce922ab17088885db610c
0e05b593844b41d5494ea
STEP 1: FROM image-registry.openshift-image-registry.svc:5000/openshift/php@s
ha256:d97466f33999951739a76bce922ab17088885db610c0e05b593844b41d5494ea
STEP 2: LABEL "io.openshift.build.commit.author"="Christian Hernandez <christ
ian.hernandez@yahoo.com>"       "io.openshift.build.commit.date"="Sun Oct 1 1
7:15:09 2017 -0700"       "io.openshift.build.commit.id"="288eda3dff43b02f7f7
b6b6b6f93396ffdf34cb2"       "io.openshift.build.commit.ref"="master"       "
io.openshift.build.commit.message"="trying to modularize"       "io.openshift
.build.source-location"="https://github.com/christianh814/openshift-php-uploa
d-demo"       "io.openshift.build.image"="image-registry.openshift-image-regi
stry.svc:5000/openshift/php@sha256:d97466f33999951739a76bce922ab17088885db610
c0e05b593844b41d5494ea"
STEP 3: ENV OPENSHIFT_BUILD_NAME="file-uploader-1"
OPENSHIFT_BUILD_NAMESP
ACE="my-shared-storage"     OPENSHIFT_BUILD_SOURCE="https://github.com/christ
ianh814/openshift-php-upload-demo"     OPENSHIFT_BUILD_COMMIT="288eda3dff43b0
2f7f7b6b6b6f93396ffdf34cb2"
STEP 4: USER root
STEP 5: COPY upload/src /tmp/src
```

```
STEP 6: RUN chown -R 1001:0 /tmp/src
STEP 7: USER 1001
STEP 8: RUN /usr/libexec/s2i/assemble
---> Installing application source...
=> sourcing 20-copy-config.sh ...
---> 17:24:39     Processing additional arbitrary httpd configuration provide
d by s2i ...
=> sourcing 00-documentroot.conf ...
=> sourcing 50-mpm-tuning.conf ...
=> sourcing 40-ssl-certs.sh ...
STEP 9: CMD /usr/libexec/s2i/run
STEP 10: COMMIT temp.builder.openshift.io/my-shared-storage/file-uploader-1:3
b83e447
Getting image source signatures

[...]
```

Push successful を確認すると、コマンドプロンプトは tail モードから復帰します。

注記

new-app コマンドは、アプリケーションを git リポジトリーから直接デプロイして、OpenShift テンプレートを使用しないため、OpenShift ルートリソースはデフォルトでは作成されません。ルートを手動で作成する必要があります。

アプリケーションのスケーリング

1. アプリケーションを 4 つのレプリカにスケーリングし、そのサービスを公開して、アプリケーションゾーンを認識して使用できるようにします。

```
$ oc expose svc/file-uploader -n my-shared-storage
```

```
$ oc scale --replicas=4 deploy/file-uploader -n my-shared-storage
```

```
$ oc get pods -o wide -n my-shared-storage
```

数分後には、4 つの file-uploader Pod が必要です。4 つの file-uploader Pod が **Running** ステータスになるまで、上記のコマンドを繰り返します。

2. PVC を作成し、これをアプリケーションに割り当てます。

```
$ oc set volume deploy/file-uploader --add --name=my-shared-storage \
-t pvc --claim-mode=ReadWriteMany --claim-size=10Gi \
--claim-name=my-shared-storage --claim-class=ocs-storagecluster-cephfs \
--mount-path=/opt/app-root/src/uploaded \
-n my-shared-storage
```

このコマンドは、以下のようになります。

- PVC を作成します。
- ボリューム定義を含めるようにアプリケーションデプロイメントを更新します。

- アプリケーションのデプロイメントを更新して、ボリュームマウントを指定されたマウントパスに割り当てます。
- 4 つのアプリケーション Pod で新規デプロイメントを作成します。

3. ボリュームの追加の結果を確認します。

```
$ oc get pvc -n my-shared-storage
```

出力例:

```
NAME                STATUS   VOLUME                                     CAPACITY   ACCESS MODES
STORAGECLASS                AGE
my-shared-storage   Bound    pvc-5402cc8a-e874-4d7e-af76-1eb05bd2e7c7   10Gi       RWX
ocs-storagecluster-cephfs   52s
```

ACCESS MODE が RWX に設定されている点に注意してください。

4 つの **file-uploader** Pod はすべて同じ RWX ボリュームを使用しています。このアクセスモードがないと、OpenShift は複数の Pod を同じ永続ボリューム (PV) に確実にアタッチしようとしません。ReadWriteOnce (RWO) PV を使用するデプロイメントをスケールアップしようとすると、Pod は同じノードに配置される可能性があります。

5.2. ゾーンを意識したデプロイメントの変更

現在、**file-uploader** Deployment はゾーンを認識せず、すべての Pod を同じゾーンでスケジュールできます。この場合、サイトに障害が発生すると、アプリケーションは利用できなくなります。詳細は、Pod トポロジー分散制約を使用した Pod 配置の制御 を参照してください。

1. アプリケーションデプロイメント設定に Pod 配置ルールを追加して、アプリケーションゾーンを認識させます。

   a. 以下のコマンドを実行して出力を確認します。

```
$ oc get deployment file-uploader -o yaml -n my-shared-storage | less
```

出力例:

```
[...]
spec:
  progressDeadlineSeconds: 600
  replicas: 4
  revisionHistoryLimit: 10
  selector:
    matchLabels:
      deployment: file-uploader
  strategy:
    rollingUpdate:
      maxSurge: 25%
      maxUnavailable: 25%
    type: RollingUpdate
  template:
    metadata:
      annotations:
        openshift.io/generated-by: OpenShiftNewApp
```

```
      creationTimestamp: null
      labels:
        deployment: file-uploader
      spec: # <-- Start inserted lines after here
        containers: # <-- End inserted lines before here
        - image: image-registry.openshift-image-registry.svc:5000/my-shared-storage/file-
uploader@sha256:a458ea62f990e431ad7d5f84c89e2fa27bdebdd5e29c5418c70c56eb81f
0a26b
          imagePullPolicy: IfNotPresent
          name: file-uploader
[...]
```

b. トポロジーゾーンラベルを使用するようにデプロイメントを編集します。

```
$ oc edit deployment file-uploader -n my-shared-storage
```

Start と **End** の間に、以下の新しい行を追加します (前のステップの出力に表示)。

```
[...]
    spec:
      topologySpreadConstraints:
        - labelSelector:
            matchLabels:
              deployment: file-uploader
          maxSkew: 1
          topologyKey: topology.kubernetes.io/zone
          whenUnsatisfiable: DoNotSchedule
        - labelSelector:
            matchLabels:
              deployment: file-uploader
          maxSkew: 1
          topologyKey: kubernetes.io/hostname
          whenUnsatisfiable: ScheduleAnyway
      nodeSelector:
        node-role.kubernetes.io/worker: ""
      containers:
[...]
```

出力例:

```
deployment.apps/file-uploader edited
```

2. デプロイを **0 個** の Pod に変更し、その後 **4 個** の Pod に戻します。これは、デプロイメントが Pod の配置に関して変更されたために必要です。

0 個の Pod へのスケールダウン

```
$ oc scale deployment file-uploader --replicas=0 -n my-shared-storage
```

出力例:

```
deployment.apps/file-uploader scaled
```

4 個の Pod へのスケールアップ

```
$ oc scale deployment file-uploader --replicas=4 -n my-shared-storage
```

出力例:

```
deployment.apps/file-uploader scaled
```

3. 4 つの Pod が datacenter1 および datacenter2 ゾーンの 4 つのノードに分散されていることを確認します。

```
$ oc get pods -o wide -n my-shared-storage | egrep '^file-uploader'| grep -v build | awk '{print $7}' | sort | uniq -c
```

出力例:

```
   1 perf1-mz8bt-worker-d2hdm
   1 perf1-mz8bt-worker-k68rv
   1 perf1-mz8bt-worker-ntkp8
   1 perf1-mz8bt-worker-qpwsr
```

使用されるゾーンラベルを検索します。

```
$ oc get nodes -L topology.kubernetes.io/zone | grep datacenter | grep -v master
```

出力例:

```
perf1-mz8bt-worker-d2hdm   Ready    worker   35d   v1.20.0+5fbfd19   datacenter1
perf1-mz8bt-worker-k68rv   Ready    worker   35d   v1.20.0+5fbfd19   datacenter1
perf1-mz8bt-worker-ntkp8   Ready    worker   35d   v1.20.0+5fbfd19   datacenter2
perf1-mz8bt-worker-qpwsr   Ready    worker   35d   v1.20.0+5fbfd19   datacenter2
```

4. ブラウザーを使用して file-uploader Web アプリケーションを使用して新規ファイルをアップロードします。

   a. 作成されたルートを検索します。

```
$ oc get route file-uploader -n my-shared-storage -o jsonpath --template="http://{.spec.host}{'\n'}"
```

   出力例:

```
http://file-uploader-my-shared-storage.apps.cluster-ocs4-abdf.ocs4-abdf.sandbox744.opentlc.com
```

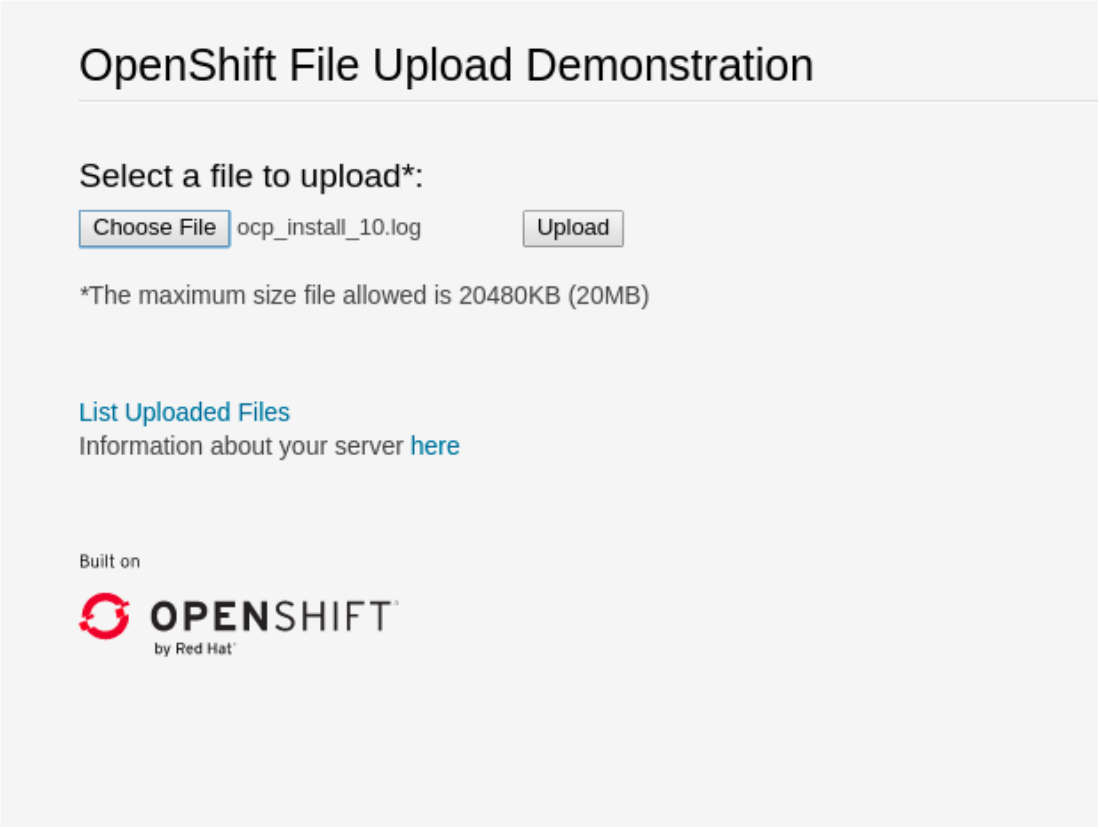
   b. 直前の手順のルートを使用して、ブラウザーを Web アプリケーションに指定します。Web アプリケーションはアップロードしたすべてのファイルを一覧表示し、新しいファイルをアップロードしたり、既存のデータをダウンロードする機能を提供します。今は何もありません。

   c. ローカルマシンから任意のファイルを選択し、これをアプリケーションにアップロードします。

      i. **Choose file** をクリックして任意のファイルを選択します。

ii. **アップロード** をクリックします。

図5.1 簡単な PHP ベースのファイルのアップロードツール



d. **List uploaded files** をクリックし、現在アップロードされているファイルの一覧を表示します。

注記

OpenShift Container Platform イメージレジストリー、Ingress ルーティング、およびモニターリングサービスはゾーンを認識しません。