Red Hat OpenShift Data Foundation 4.15

OpenShift ワークロード用の OpenShift Data Foundation Disaster Recovery の設定

ストレッチクラスターを使用した Disaster Recovery など、大都市圏および地方地域向けの OpenShift Data Foundation Disaster Recovery 機能が一般利用可能になりました。

Last Updated: 2024-04-25

Red Hat OpenShift Data Foundation 4.15 OpenShift ワークロード用の OpenShift Data Foundation Disaster Recovery の設定

ストレッチクラスターを使用した Disaster Recovery など、大都市圏および地方地域向けの OpenShift Data Foundation Disaster Recovery 機能が一般利用可能になりました。

法律上の通知

概要

このソリューションガイドの目的は、Advanced Cluster Management を使用した災害復旧のために OpenShift Data Foundation をデプロイし、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するストレッチクラスターに必要な手順を詳しく説明することです。

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)

Red Hat ドキュメントに対する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。

フィードバックを送信するには、Bugzilla チケットを作成します。

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 OPENSHIFT DATA FOUNDATION 災害復旧の概要

障害復旧 (DR) は、自然災害または人為的災害からビジネスクリティカルなアプリケーションを回復し、継続する機能です。これは、重大な有害事象の発生時に事業の継続性を維持するために設計されている、主要な組織における事業継続ストラテジー全体の設定要素です。

OpenShift Data Foundation の DR 機能は、複数の Red Hat OpenShift Container Platform クラスターにわたる DR を可能にし、次のように分類されます。

- **Metro-DR**
  Metro-DR は、データセンターが利用できない場合でも、データを失うことなくビジネスの継続性を保証します。パブリッククラウドでは、これらはアベイラビリティーゾーンの障害からの保護に似ています。

- **Regional-DR**
  Regional-DR は、地理的な地域が利用できない場合でもビジネス継続性を確保し、予測可能な量のデータの損失を受け入れます。パブリッククラウドでは、これはリージョンの障害からの保護と似ています。

- **ストレッチクラスターを使用した障害復旧**
  ストレッチクラスターソリューションは、2 つのデータセンター (レイテンシーが低いノードおよび arbiter ノード 1 つ) にまたがる単一の OpenShift クラスターにおいて OpenShift Data Foundation ベースの同期レプリケーションを行い、データの損失なしに障害復旧を実現することでビジネスの継続性を確保します。

Metro-DR のゾーン障害と Regional-DR のリージョン障害は、通常、**目標復旧時点 (RPO)** と **目標復旧時間 (RTO)** という用語を使用して表現されます。

- **RPO** は、永続データのバックアップまたはスナップショットを作成する頻度の尺度です。実際には、RPO は、停止後に失われるか、再入力する必要があるデータの量を示します。

- **RTO** は、企業が許容できるダウンタイムの量です。RTO は、ビジネスの中断が通知されてから、システムが回復するまでにどれくらいかかるか? という質問に答えます。

このガイドの目的は、ある OpenShift Container Platform クラスターから別のクラスターにアプリケーションをフェイルオーバーし、同じアプリケーションを元のプライマリークラスターに再配置するために必要な災害復旧の手順とコマンドについて詳しく説明することです。

第2章 障害復旧サブスクリプションの要件

Red Hat OpenShift Data Foundation でサポートされる障害復旧機能では、障害復旧ソリューションを正常に実装するために以下の前提条件をすべて満たす必要があります。

- 有効な Red Hat OpenShift Data Foundation Advanced エンタイトルメント
- 有効な Red Hat Advanced Cluster Management for Kubernetes サブスクリプション

ソースまたは宛先としてアクティブレプリケーションに参加している PV を含む Red Hat OpenShift Data Foundation クラスターには、OpenShift Data Foundation Advanced エンタイトルメントが必要です。このサブスクリプションは、ソースクラスターと宛先クラスターの両方でアクティブにする必要があります。

OpenShift Data Foundation のサブスクリプションの仕組みを確認するには、OpenShift Data Foundation subscriptions に関するナレッジベースの記事 を参照してください。

第3章 OPENSHIFT DATA FOUNDATION 向けの METRO-DR ソリューション

本ガイドのこのセクションでは、アプリケーションを OpenShift Container Platform クラスターから別のクラスターにフェイルオーバーし、そのアプリケーションを元のプライマリークラスターにフォールバックできるようにするために必要な Metro Disaster Recovery (Metro DR) の手順とコマンドを詳しく説明します。この場合、OpenShift Container Platform クラスターは Red Hat Advanced Cluster Management (RHACM) を使用して作成またはインポートされ、OpenShift Container Platform クラスター間の距離は 10 ミリ秒未満の RTT レイテンシーに制限されます。

アプリケーションの永続ストレージは、2 つの場所に拡張された外部 Red Hat Ceph Storage (RHCS) クラスターによって提供され、OpenShift Container Platform インスタンスはこのストレージクラスターに接続されます。サイトが停止した場合に RHCS クラスターのクォーラムを確立するには、3 番目の場所 (OpenShift Container Platform インスタンスがデプロイされている場所とは異なる場所) にストレージモニターサービスを備えた arbiter ノードが必要になります。この 3 番目の場所は、OpenShift Container Platform インスタンスに接続されたストレージクラスターから約 100 ミリ秒の RTT の範囲内に配置できます。

以下は、2 つの異なる OpenShift Container Platform クラスター間で、OpenShift Data Foundation および RHACM を使用して、OpenShift Disaster Recovery (ODR) 機能を設定および実行するために必要な Metro DR 手順の概要です。マネージドクラスターと呼ばれるこれら 2 つのクラスターに加えて、Red Hat Advanced Cluster Management (RHACM) ハブクラスターとなる 3 つ目の OpenShift Container Platform クラスターが必要です。

重要

OpenShift Data Foundation を使用して、OpenShift 仮想化テクノロジーに基づいたワークロード用の Metro 災害復旧ソリューションを簡単に設定できるようになりました。詳細は、ナレッジベースの記事 を参照してください。

3.1. METRO-DR ソリューションのコンポーネント

Metro-DR は、Red Hat Advanced Cluster Management for Kubernetes、Red Hat Ceph Storage、および OpenShift Data Foundation コンポーネントで設定され、OpenShift Container Platform クラスター全体でアプリケーションとデータのモビリティを提供します。

Red Hat Advanced Cluster Management for Kubernetes

Red Hat Advanced Cluster Management (RHACM) は、複数のクラスターとアプリケーションのライフサイクルを管理する機能を提供します。したがって、マルチクラスター環境でのコントロールプレーンとして機能します。

RHACM は 2 つの部分に分かれています。

- RHACM Hub: マルチクラスターコントロールプレーンで実行されるコンポーネント
- マネージドクラスター: マネージドクラスターで実行されるコンポーネント

この製品の詳細については、RHACM のドキュメント および RHACM のアプリケーションの管理 を参照してください。

Red Hat Ceph Storage

Red Hat Ceph Storage は、非常にスケーラブルでオープンなソフトウェア定義のストレージプラットフォームであり、最も安定したバージョンの Ceph ストレージシステムと Ceph 管理プラットフォーム、デプロイメントユーティリティー、およびサポートサービスを組み合わせたものです。エンタープ

ライズデータの保存コストを大幅に削減し、組織が指数関数的なデータの成長を管理できるようにします。このソフトウェアは、パブリックまたはプライベートのクラウドデプロイメント向けの堅牢かつ最新のペタバイトスケールのストレージプラットフォームです。

詳細な製品情報については、Red Hat Ceph Storage を参照してください。

OpenShift Data Foundation

OpenShift Data Foundation は、OpenShift Container Platform クラスターでステートフルなアプリケーション用のストレージをプロビジョニングし、管理する機能を提供します。これは、OpenShift Data Foundation コンポーネントスタックで Rook によって管理されるストレージプロバイダーとして Ceph によってサポートされ、Ceph-CSI はステートフルアプリケーションの永続ボリュームのプロビジョニングおよび管理を行います。

OpenShift DR

OpenShift DR は、RHACM を使用してデプロイおよび管理される一連のピア OpenShift クラスター全体のステートフルアプリケーションの障害復旧オーケストレーターであり、永続ボリュームでのアプリケーションの状態のライフサイクルのオーケストレーションを行うためのクラウドネイティブインターフェイスを提供します。これには以下が含まれます。

- OpenShift クラスター全体でアプリケーションとその状態関係を保護する
- アプリケーションとその状態をピアクラスターにフェイルオーバーする
- アプリケーションとその状態を以前にデプロイされたクラスターに再配置する

OpenShift DR は 3 つのコンポーネントに分類されます。

- **ODF マルチクラスターオーケストレーター**: マルチクラスターコントロールプレーン (RHACM ハブ) にインストールされ、メトロおよびリージョナル DR 関係のために OpenShift Data Foundation クラスターの設定とピアリングを調整します。
- **OpenShift DR Hub Operator**: ハブクラスターに ODF Multicluster Orchestrator インストールの一部として自動的にインストールされ、DR 対応アプリケーションのフェイルオーバーまたは再配置を調整します。
- **OpenShift DR Cluster Operator**: Metro および Regional DR 関係の一部である各マネージドクラスターに自動的にインストールされ、アプリケーションのすべての PVC のライフサイクルを管理します。

3.2. METRO-DR デプロイメントワークフロー

このセクションでは、最新バージョンの Red Hat OpenShift Data Foundation、Red Hat Ceph Storage (RHCS)、および Red Hat Advanced Cluster Management for Kubernetes (RHACM) バージョン 2.10 以降を使用して、2 つの異なる OpenShift Container Platform クラスターにわたって Metro-DR 機能を設定およびデプロイするために必要な手順の概要を示します。.2 つのマネージドクラスターに加えて、Advanced Cluster Management をデプロイするのに、3 つ目の OpenShift Container Platform クラスターが必要です。

インフラストラクチャーを設定するには、指定した順序で以下の手順を実行します。

1. DR ソリューションの一部であるハブ、プライマリー、およびセカンダリー Openshift Container Platform クラスターの要件が満たされていることを確認します。Metro-DR を有効にするための要件 を参照してください。

2. arbiter を使用して Red Hat Ceph Storage ストレッチクラスターをデプロイするための要件を満たしていることを確認してください。Red Hat Ceph Storage の導入の要件 を参照してください。

3. Red Hat Ceph Storage ストレッチモードをデプロイして設定します。拡張モード機能を使用して 2 つの異なるデータセンターで Ceph クラスターを有効にする手順については、Red Hat Ceph Storage のデプロイ を参照してください。

4. OpenShift Data Foundation operator をインストールし、プライマリーおよびセカンダリーのマネージドクラスターにストレージシステムを作成します。マネージドクラスターへの OpenShift Data Foundation のインストール を参照してください。

5. ハブクラスターに ODF マルチクラスターオーケストレーターをインストールします。ハブクラスターへの ODF Multicluster Orchestrator のインストール を参照してください。

6. ハブ、プライマリー、およびセカンダリークラスター間の SSL アクセスを設定します。クラスター間での SSL アクセスの設定 を参照してください。

7. プライマリークラスターとセカンダリークラスター全体で DR 保護を必要とするアプリケーションで使用する DRPolicy リソースを作成します。Creating Disaster Recovery Policy on Hub cluster を参照してください。

注記

Metro-DR ソリューションには、DRpolicy を 1 つだけ指定できます。

8. 以下を使用して災害復旧ソリューションをテストします。

   a. **サブスクリプションベース** のアプリケーション:

      - サンプルアプリケーションを作成します。サンプルアプリケーションの作成 を参照してください。
      - マネージドクラスター間でサンプルアプリケーションを使用して、フェイルオーバーと再配置操作をテストします。サブスクリプションベースのアプリケーションのフェイルオーバー と サブスクリプションベースのアプリケーションの再配置 を参照してください。

   b. **ApplicationSet ベース** のアプリケーション:

      - サンプルアプリケーションを作成します。ApplicationSet ベースのアプリケーションの作成 を参照してください。
      - マネージドクラスター間でサンプルアプリケーションを使用して、フェイルオーバーと再配置操作をテストします。ApplicationSet ベースのアプリケーションのフェイルオーバー と ApplicationSet ベースのアプリケーションの再配置 を参照してください。

3.3. METRO-DR を有効にする要件

Red Hat OpenShift Data Foundation でサポートされる障害復旧ソリューションをインストールするための前提条件は次のとおりです。

- 相互にネットワーク接続が可能な以下の OpenShift クラスターが必要です。
  - Red Hat Advanced Cluster Management (RHACM) for Kubernetes operator がインストールされている **ハブクラスター**。

- OpenShift Data Foundation が実行されている **プライマリーマネージドクラスター**。
- OpenShift Data Foundation が実行されている **セカンダリーマネージドクラスター**。

注記

ハブリカバリーセットアップを設定するには、パッシブハブとして機能する 4 番目のクラスターが必要です。プライマリー管理対象クラスター (Site-1) はアクティブ RHACM ハブクラスターと共存でき、パッシブハブクラスターはセカンダリー管理対象クラスター (Site-2) とともに配置できます。あるいは、アクティブな RHACM ハブクラスターを、サイト 1 のプライマリー管理対象クラスターまたはサイト 2 のセカンダリークラスターのいずれかの障害の影響を受けない中立サイト (サイト 3) に配置することもできます。この状況では、パッシブハブクラスターを使用する場合は、サイト 2 のセカンダリークラスターと一緒に配置できます。詳細は、ハブリカバリー用のパッシブハブクラスターの設定 を参照してください。

ハブリカバリーはテクノロジープレビュー機能で、テクノロジープレビューのサポート制限の対象となります。

- RHACM Operator と Multiclusterhub がハブクラスターにインストールされていることを確認します。手順については、RHACM インストールガイド を参照してください。
  Operator が正常にインストールされると、Web console update is available というメッセージを含むポップアップがユーザーインターフェイスに表示されます。このポップアップから **Refresh web console** をクリックして、反映するコンソールを変更します。

重要

アプリケーショントラフィックのルーティングとリダイレクトが適切に設定されていることを確認してください。

- ハブクラスターで以下を行います。
  - **All Clusters → Infrastructure → Clusters** に移動します。
  - RHACM コンソールを使用して、**プライマリーマネージドクラスター** および **セカンダリーマネージドクラスター** をインポートまたは作成します。
  - 環境に適したオプションを選択します。

  マネージドクラスターが正常に作成またはインポートされると、コンソールでインポートまたは作成されたクラスターのリストを確認できます。手順は、クラスターの作成 および ハブクラスターへのターゲット管理対象クラスターのインポート を参照してください。

警告

Openshift Container Platform マネージドクラスターと Red Hat Ceph Storage (RHCS) ノードには距離制限があります。サイト間のネットワーク遅延は、10 ミリ秒の往復時間 (RTT) 未満である必要があります。

3.4. ARBITER を使用して RED HAT CEPH STORAGE ストレッチクラスターをデプロイするための要件

Red Hat Ceph Storage は、標準的で経済的なサーバーおよびディスク上で統一されたソフトウェア定義ストレージを提供するオープンソースエンタープライズプラットフォームです。ブロック、オブジェクト、ファイルストレージを 1 つのプラットフォームに統合することで、Red Hat Ceph Storage はすべてのデータを効率的かつ自動的に管理します。そのため、それを使用するアプリケーションおよびワークロードに集中することができます。

このセクションでは、Red Hat Ceph Storage デプロイメントの基本的な概要を説明します。より複雑なデプロイメントは、Red Hat Ceph Storage 7 の公式ドキュメントガイド を参照してください。

注記

劣化すると **min_size=1** で実行されるため、Flash メディアのみがサポートされています。ストレッチモードは、オールフラッシュ OSD でのみ使用してください。オールフラッシュ OSD を使用すると、接続が復元された後の回復に必要な時間が最小限に抑えられるため、データ損失の可能性が最小限に抑えられます。

重要

イレイジャーコーディングされたプールは、ストレッチモードでは使用できません。

3.4.1. ハードウェア要件

Red Hat Ceph Storage をデプロイするためのハードウェアの最小要件については、コンテナー化された Ceph の最小ハードウェア推奨事項 を参照してください。

表3.1 Red Hat Ceph Storage クラスターデプロイメントの物理サーバーの場所と Ceph コンポーネントのレイアウト:

ノード名	データセンター	Ceph コンポーネント
ceph1	DC1	OSD+MON+MGR
ceph2	DC1	OSD+MON
ceph3	DC1	OSD+MDS+RGW
ceph4	DC2	OSD+MON+MGR
ceph5	DC2	OSD+MON
ceph6	DC2	OSD+MDS+RGW
ceph7	DC3	MON

3.4.2. ソフトウェア要件

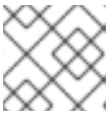
Red Hat Ceph Storage 6.7の最新のソフトウェアバージョンを使用してください。

Red Hat Ceph Storage でサポートされているオペレーティングシステムのバージョンの詳細については、Red Hat Ceph Storage: Supported configurations に関するナレッジベースの記事を参照してください。

3.4.3. ネットワーク設定要件

推奨される Red Hat Ceph Storage 設定は次のとおりです。

- パブリックネットワークとプライベートネットワークを 1 つずつ、合計 2 つのネットワークが必要です。
- すべてのデータセンターの Cephs プライベートネットワークとパブリックネットワークの VLAN とサブネットをサポートする 3 つの異なるデータセンターが必要です。



注記

データセンターごとに異なるサブネットを使用できます。

- Red Hat Ceph Storage Object Storage Devices (OSD) を実行している 2 つのデータセンター間のレイテンシーは、RTT で 10 ミリ秒を超えることはできません。**arbiter** データセンターの場合、これは、他の 2 つの OSD データセンターに対して最大 100 ミリ秒の RTT という高い値でテストされました。

このガイドで使用した基本的なネットワーク設定の例を次に示します。

- **DC1: Ceph パブリック/プライベートネットワーク**:10.0.40.0/24
- **DC2: Ceph パブリック/プライベートネットワーク**:10.0.40.0/24
- **DC3: Ceph パブリック/プライベートネットワーク**:10.0.40.0/24

必要なネットワーク環境の詳細は、Ceph ネットワーク設定 を参照してください。

3.5. RED HAT CEPH STORAGE の導入

3.5.1. ノードのデプロイ前の手順

Red Hat Ceph Storage Ceph クラスターをインストールする前に、必要なすべての要件を満たすために、以下の手順を実施します。

1. 全ノードを Red Hat Network または Red Hat Satellite に登録し、有効なプールにサブスクライブします。

```
subscription-manager register
subscription-manager subscribe --pool=8a8XXXXXX9e0
```

2. 次のリポジトリーの Ceph クラスター内のすべてのノードへのアクセスを有効にします。
   - **rhel9-for-x86_64-baseos-rpms**
   - **rhel9-for-x86_64-appstream-rpms**

```
subscription-manager repos --disable="*" --enable="rhel9-for-x86_64-baseos-rpms" --
enable="rhel9-for-x86_64-appstream-rpms"
```

3. オペレーティングシステムの RPM を最新バージョンに更新し、必要に応じて再起動します。

```
dnf update -y
reboot
```

4. クラスターからノードを選択して、ブートストラップノードにします。**ceph1** は、この例の今後のブートストラップノードです。
ブートストラップノード **ceph1** でのみ、**ansible-2.9-for-rhel-9-x86_64-rpms** および **rhceph-6-tools-for-rhel-9-x86_64-rpms** リポジトリーを有効にします。

```
subscription-manager repos --enable="ansible-2.9-for-rhel-9-x86_64-rpms" --
enable="rhceph-6-tools-for-rhel-9-x86_64-rpms"
```

5. すべてのホストでベア/短縮ホスト名を使用して **hostname** を設定します。

```
hostnamectl set-hostname <short_name>
```

6. Red Hat Ceph Storage を使用して Red Hat Ceph Storage をデプロイするためのホスト名設定を確認します。

```
$ hostname
```

出力例:

```
ceph1
```

7. /etc/hosts ファイルを変更し、DNS ドメイン名を使用して DOMAIN 変数を設定して、fqdn エントリーを 127.0.0.1IP に追加します。

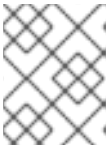
```
DOMAIN="example.domain.com"

cat <<EOF >/etc/hosts
127.0.0.1 $(hostname).${DOMAIN} $(hostname) localhost localhost.localdomain localhost4
localhost4.localdomain4
::1       $(hostname).${DOMAIN} $(hostname) localhost6 localhost6.localdomain6
EOF
```

8. **hostname -f** オプションを使用して、**fqdn** の長いホスト名を確認します。

```
$ hostname -f
```

出力例:

```
ceph1.example.domain.com
```



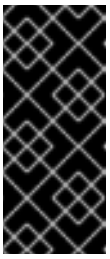
注記

これらの変更が必要な理由の詳細については、完全修飾ドメイン名とベアホスト名 を参照してください。

9. ブートストラップノードで次の手順を実行します。この例では、ブートストラップノードは **ceph1** です。

a. **cephadm-ansible** RPM パッケージをインストールします。

```
$ sudo dnf install -y cephadm-ansible
```



重要

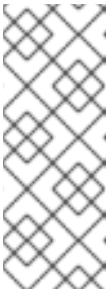
Ansible Playbook を実行するには、Red Hat Ceph Storage クラスターに設定されているすべてのノードに **ssh** パスワードなしでアクセスできる必要があります。設定されたユーザー (たとえば、**deployment-user**) が、パスワードを必要とせずに **sudo** コマンドを呼び出すための root 権限を持っていることを確認してください。

b. カスタムキーを使用するには、選択したユーザー (たとえば、**deployment-user**) の ssh 設定ファイルを設定して、ssh 経由でノードに接続するために使用される ID/キーを指定します。

```
cat <<EOF > ~/.ssh/config
Host ceph*
   User deployment-user
   IdentityFile ~/.ssh/ceph.pem
EOF
```

c. Ansible インベントリーを構築します。

```
cat <<EOF > /usr/share/cephadm-ansible/inventory
ceph1
ceph2
ceph3
ceph4
ceph5
ceph6
ceph7
[admin]
ceph1
ceph4
EOF
```



注記

ここでは、2 つの異なるデータセンターに属するホスト (**Ceph1** および **Ceph4**) が、インベントリーファイルの [admin] グループの一部として設定され、**cephadm** によって **_admin** としてタグ付けされています。これらの各管理ノードは、ブートストラッププロセス中に admin ceph キーリングを受信するため、1 つのデータセンターがダウンしたときに、他の使用可能な管理ノードを使用して確認できます。

d. プリフライト Playbook を実行する前に、**ansible** が ping モジュールを使用してすべてのノードにアクセスできることを確認します。

```
$ ansible -i /usr/share/cephadm-ansible/inventory -m ping all -b
```

出力例:

```
ceph6 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/libexec/platform-python"
    },
    "changed": false,
    "ping": "pong"
}
ceph4 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/libexec/platform-python"
    },
    "changed": false,
    "ping": "pong"
}
ceph3 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/libexec/platform-python"
    },
    "changed": false,
    "ping": "pong"
}
ceph2 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/libexec/platform-python"
    },
    "changed": false,
    "ping": "pong"
}
ceph5 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/libexec/platform-python"
    },
    "changed": false,
    "ping": "pong"
}
ceph1 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/libexec/platform-python"
    },
    "changed": false,
    "ping": "pong"
}
ceph7 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/libexec/platform-python"
    },
    "changed": false,
    "ping": "pong"
}
```

e. **/usr/share/cephadm-ansible** ディレクトリーに移動します。

f. ansible-playbook を、相対ファイルパスを指定して実行します。

```
$ ansible-playbook -i /usr/share/cephadm-ansible/inventory /usr/share/cephadm-ansible/cephadm-preflight.yml --extra-vars "ceph_origin=rhcs"
```

プリフライト Playbook は RHCS **dnf** リポジトリーを設定し、ブートストラップ用にストレージクラスターを準備します。また、podman、lvm2、chronyd、および cephadm もインストールします。**cephadm-ansible** および **cephadm-** preflight.yml のデフォルトの場所は **/usr/share/cephadm-ansible** です。詳細は、プリフライト Playbook の実行 を参照してください。

3.5.2. Cephadm ユーティリティーを使用したクラスターのブートストラップとサービスのデプロイメント

cephadm ユーティリティーは、cephadm ブートストラップコマンドが実行されているローカルノード上に、新しい Red Hat Ceph Storage クラスターの単一の Ceph Monitor デーモンと Ceph Manager デーモンをインストールし、開始します。

このガイドでは、クラスター仕様の yaml ファイルを使用して、クラスターをブートストラップし、必要なすべての Red Hat Ceph Storage サービスをワンステップでデプロイします。

展開中に問題が見つかった場合は、展開を 2 つの手順に分割することで、エラーのトラブルシューティングが容易になる場合があります。

1. ブートストラップ
2. サービスの展開

注記

ブートストラッププロセスの詳細は、新規ストレージクラスターのブートストラップ を参照してください。

手順

1. 次のように、json ファイルを使用してコンテナーレジストリーに対して認証を行うための json ファイルを作成します。

```
$ cat <<EOF > /root/registry.json
{
 "url":"registry.redhat.io",
 "username":"User",
 "password":"Pass"
}
EOF
```

2. Red Hat Ceph Storage クラスターにノードを追加する **cluster-spec.yaml** を作成し、表 3.1 に従ってサービスを実行する場所に特定のラベルを設定します。

```
cat <<EOF > /root/cluster-spec.yaml
service_type: host
addr: 10.0.40.78  ## <XXX.XXX.XXX.XXX>
hostname: ceph1   ##  <ceph-hostname-1>
location:
  root: default
  datacenter: DC1
```

```
labels:
  - osd
  - mon
  - mgr
---
service_type: host
addr: 10.0.40.35
hostname: ceph2
location:
  datacenter: DC1
labels:
  - osd
  - mon
---
service_type: host
addr: 10.0.40.24
hostname: ceph3
location:
  datacenter: DC1
labels:
  - osd
  - mds
  - rgw
---
service_type: host
addr: 10.0.40.185
hostname: ceph4
location:
  root: default
  datacenter: DC2
labels:
  - osd
  - mon
  - mgr
---
service_type: host
addr: 10.0.40.88
hostname: ceph5
location:
  datacenter: DC2
labels:
  - osd
  - mon
---
service_type: host
addr: 10.0.40.66
hostname: ceph6
location:
  datacenter: DC2
labels:
  - osd
  - mds
  - rgw
---
service_type: host
addr: 10.0.40.221
```

```
hostname: ceph7
labels:
  - mon
---
service_type: mon
placement:
  label: "mon"
---
service_type: mds
service_id: cephfs
placement:
  label: "mds"
---
service_type: mgr
service_name: mgr
placement:
  label: "mgr"
---
service_type: osd
service_id: all-available-devices
service_name: osd.all-available-devices
placement:
  label: "osd"
spec:
  data_devices:
    all: true
---
service_type: rgw
service_id: objectgw
service_name: rgw.objectgw
placement:
  count: 2
  label: "rgw"
spec:
  rgw_frontend_port: 8080
EOF
```

3. ブートストラップノードから設定された Red Hat Ceph Storage パブリックネットワークで NIC の IP を取得します。**10.0.40.0** を ceph パブリックネットワークで定義したサブネットに置き換えた後、次のコマンドを実行します。

```
$ ip a | grep 10.0.40
```

出力例:

```
10.0.40.78
```

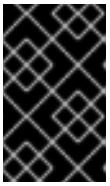
4. クラスター内の最初の Monitor ノードとなるノードで、**root** ユーザーとして **Cephadm** bootstrap コマンドを実行します。**IP_ADDRESS** オプションは、**cephadm bootstrap** コマンドの実行に使用しているノードの IP アドレスです。

注記

パスワードなしの SSH アクセス用に **root** ではなく別のユーザーを設定した場合は、**cepadm bootstrap** コマンドで **--ssh-user=** フラグを使用します。

default/id_rsa ssh キー名以外を使用している場合は、**cephadm** コマンドで **--ssh-private-key** および **--ssh-public-key** オプションを使用します。

```
$ cephadm  bootstrap --ssh-user=deployment-user --mon-ip 10.0.40.78 --apply-spec
/root/cluster-spec.yaml --registry-json /root/registry.json
```



重要

ローカルノードが完全修飾ドメイン名 (FQDN) を使用する場合は、コマンドラインで **--allow-fqdn-hostname** オプションを **cephadm bootstrap** に追加します。

ブートストラップが終了すると、前の cephadm bootstrap コマンドから次の出力が表示されます。

```
You can access the Ceph CLI with:

 sudo /usr/sbin/cephadm shell --fsid dd77f050-9afe-11ec-a56c-029f8148ea14 -c
/etc/ceph/ceph.conf -k /etc/ceph/ceph.client.admin.keyring

Consider enabling telemetry to help improve Ceph:

 ceph telemetry on

For more information see:

 https://docs.ceph.com/docs/pacific/mgr/telemetry/
```

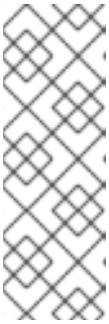
5. ceph1 の Ceph CLI クライアントを使用して、Red Hat Ceph Storage クラスターデプロイメントのステータスを確認します。

```
$ ceph -s
```

出力例:

```
cluster:
  id:     3a801754-e01f-11ec-b7ab-005056838602
  health: HEALTH_OK

services:
  mon: 5 daemons, quorum ceph1,ceph2,ceph4,ceph5,ceph7 (age 4m)
  mgr: ceph1.khuuot(active, since 5m), standbys: ceph4.zotfsp
  osd: 12 osds: 12 up (since 3m), 12 in (since 4m)
  rgw: 2 daemons active (2 hosts, 1 zones)

data:
  pools:   5 pools, 107 pgs
```

```
objects: 191 objects, 5.3 KiB
usage:   105 MiB used, 600 GiB / 600 GiB avail
         105 active+clean
```



注記

すべてのサービスが開始されるまでに数分かかる場合があります。

OSD が設定されていないときに、グローバルリカバリーイベントが発生するのは正常です。

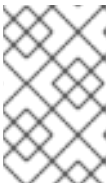
ceph orch ps および **ceph orch ls** を使用して、サービスのステータスをさらに確認できます。

6. すべてのノードが **cephadm** クラスターの一部であるかどうかを確認します。

```
$ ceph orch host ls
```

出力例:

```
HOST   ADDR         LABELS  STATUS
ceph1  10.0.40.78   _admin osd mon mgr
ceph2  10.0.40.35   osd mon
ceph3  10.0.40.24   osd mds rgw
ceph4  10.0.40.185   osd mon mgr
ceph5  10.0.40.88   osd mon
ceph6  10.0.40.66   osd mds rgw
ceph7  10.0.40.221   mon
```



注記

ceph1 は [admin] グループの一部として **cephadm-ansible** インベントリーで設定されているため、ホストから Ceph コマンドを直接実行できます。Ceph 管理キーは、**cephadm bootstrap** プロセス中にホストにコピーされました。

7. データセンターでの Ceph モニターサービスの現在の配置を確認します。

```
$ ceph orch ps | grep mon | awk '{print $1 " " $2}'
```

出力例:

```
mon.ceph1 ceph1
mon.ceph2 ceph2
mon.ceph4 ceph4
mon.ceph5 ceph5
mon.ceph7 ceph7
```

8. データセンターでの Ceph 管理サービスの現在の配置を確認します。

```
$ ceph orch ps | grep mgr | awk '{print $1 " " $2}'
```

出力例:

```
mgr.ceph2.ycgwyz ceph2
mgr.ceph5.kremtt ceph5
```

9. ceph osd クラッシュマップレイアウトをチェックして、各ホストに 1 つの OSD が設定され、そのステータスが **UP** であることを確認します。また、表 3.1 で指定されているように、各ノードが適切なデータセンターバケットの下にあることを再確認してください。

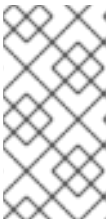
```
$ ceph osd tree
```

出力例:

```
ID   CLASS  WEIGHT   TYPE NAME         STATUS  REWEIGHT  PRI-AFF
-1         0.87900  root default
-16        0.43950      datacenter DC1
-11        0.14650          host ceph1
 2    ssd  0.14650              osd.2       up   1.00000  1.00000
 -3        0.14650          host ceph2
 3    ssd  0.14650              osd.3       up   1.00000  1.00000
-13        0.14650          host ceph3
 4    ssd  0.14650              osd.4       up   1.00000  1.00000
-17        0.43950      datacenter DC2
 -5        0.14650          host ceph4
 0    ssd  0.14650              osd.0       up   1.00000  1.00000
 -9        0.14650          host ceph5
 1    ssd  0.14650              osd.1       up   1.00000  1.00000
 -7        0.14650          host ceph6
 5    ssd  0.14650              osd.5       up   1.00000  1.00000
```

10. 新しい RDB ブロックプールを作成して有効にします。

```
$ ceph osd pool create 32 32
$ ceph osd pool application enable rbdpool rbd
```



注記

コマンドの最後にある 32 という数字は、このプールに割り当てられている PG の数です。PG の数は、クラスター内の OSD の数、プールの予想使用率など、いくつかの要因によって異なります。次の計算機を使用して、必要な PG の数を決定できます。プール計算機ごとの Ceph 配置グループ (PG)。

11. RBD プールが作成されたことを確認します。

```
$ ceph osd lspools | grep rbdpool
```

出力例:

```
 3 rbdpool
```

12. MDS サービスがアクティブであり、各データセンターに 1 つのサービスが配置されていることを確認します。

```
$ ceph orch ps | grep mds
```

出力例:

```
mds.cephfs.ceph3.cjpbqo    ceph3           running (17m)   117s ago  17m    16.1M       -
16.2.9
mds.cephfs.ceph6.lqmgqt    ceph6           running (17m)   117s ago  17m    16.1M       -
16.2.9
```

13. CephFS ボリュームを作成します。

```
$ ceph fs volume create cephfs
```

注記

ceph fs volume create コマンドは、必要なデータとメタ CephFS プールも作成します。詳細は、Ceph ファイルシステムの設定とマウント を参照してください。

14. **Ceph** のステータスを確認して、MDS デーモンがどのようにデプロイされたかを確認します。状態がアクティブで、**ceph6** がこのファイルシステムのプライマリー MDS で、**ceph3** がセカンダリー MDS であることを確認します。

```
$ ceph fs status
```

出力例:

```
cephfs - 0 clients
======
RANK  STATE         MDS            ACTIVITY     DNS    INOS   DIRS   CAPS
 0    active  cephfs.ceph6.ggjywj  Reqs:    0 /s    10     13     12      0
       POOL          TYPE     USED  AVAIL
cephfs.cephfs.meta  metadata  96.0k   284G
cephfs.cephfs.data    data       0    284G
   STANDBY MDS
cephfs.ceph3.ogcqkl
```

15. RGW サービスがアクティブであることを確認します。

```
$ ceph orch ps | grep rgw
```

出力例:

```
rgw.objectgw.ceph3.kkmxgb  ceph3  *:8080       running (7m)      3m ago   7m    52.7M       -
16.2.9
rgw.objectgw.ceph6.xmnpah  ceph6  *:8080       running (7m)      3m ago   7m    53.3M       -
16.2.9
```

3.5.3. Red Hat Ceph Storage ストレッチモードの設定

cephadm を使用して Red Hat Ceph Storage クラスターが完全にデプロイされたら、次の手順でストレッチクラスターモードを設定します。新しいストレッチモードは、2 サイトのケースを処理するように設計されています。

手順

1. ceph mon dump コマンドを使用して、モニターが使用している現在の選挙戦略を確認します。ceph クラスターのデフォルトでは、接続はクラシックに設定されています。

```
ceph mon dump | grep election_strategy
```

出力例:

```
dumped monmap epoch 9
election_strategy: 1
```

2. モニターの選択を接続に変更します。

```
ceph mon set election_strategy connectivity
```

3. 前の cephmondump コマンドを再度実行して、election_strategy 値を確認します。

```
$ ceph mon dump | grep election_strategy
```

出力例:

```
dumped monmap epoch 10
election_strategy: 3
```

さまざまな選択戦略の詳細は、モニター選出ストラテジーの設定 を参照してください。

4. すべての Ceph モニターの場所を設定します。

```
ceph mon set_location ceph1 datacenter=DC1
ceph mon set_location ceph2 datacenter=DC1
ceph mon set_location ceph4 datacenter=DC2
ceph mon set_location ceph5 datacenter=DC2
ceph mon set_location ceph7 datacenter=DC3
```

5. 各モニターに適切な場所があることを確認します。

```
$ ceph mon dump
```

出力例:

```
epoch 17
fsid dd77f050-9afe-11ec-a56c-029f8148ea14
last_changed 2022-03-04T07:17:26.913330+0000
created 2022-03-03T14:33:22.957190+0000
min_mon_release 16 (pacific)
election_strategy: 3
0: [v2:10.0.143.78:3300/0,v1:10.0.143.78:6789/0] mon.ceph1; crush_location
{datacenter=DC1}
1: [v2:10.0.155.185:3300/0,v1:10.0.155.185:6789/0] mon.ceph4; crush_location
{datacenter=DC2}
2: [v2:10.0.139.88:3300/0,v1:10.0.139.88:6789/0] mon.ceph5; crush_location
{datacenter=DC2}
3: [v2:10.0.150.221:3300/0,v1:10.0.150.221:6789/0] mon.ceph7; crush_location
```

```
{datacenter=DC3}
4: [v2:10.0.155.35:3300/0,v1:10.0.155.35:6789/0] mon.ceph2; crush_location
{datacenter=DC1}
```

6. **crushtool** コマンドを使用するために **ceph-base** RPM パッケージをインストールして、この OSD クラッシュトポロジーを利用する CRUSH ルールを作成します。

```
$ dnf -y install ceph-base
```

CRUSH ルールセットの詳細は、Ceph CRUSH ルールセット を参照してください。

7. コンパイルされた CRUSH マップをクラスターから取得します。

```
$ ceph osd getcrushmap > /etc/ceph/crushmap.bin
```

8. CRUSH マップを逆コンパイルし、これをテキストファイルに変換して編集できるようにします。

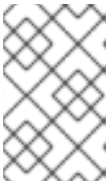
```
$ crushtool -d /etc/ceph/crushmap.bin -o /etc/ceph/crushmap.txt
```

9. ファイルの末尾にあるテキストファイル **/etc/ceph/crushmap.txt** を編集して、以下のルールを CRUSH マップに追加します。

```
$ vim /etc/ceph/crushmap.txt
```

```
rule stretch_rule {
        id 1
        type replicated
        min_size 1
        max_size 10
        step take default
        step choose firstn 0 type datacenter
        step chooseleaf firstn 2 type host
        step emit
}
# end crush map
```

この例は、両方の OpenShift Container Platform クラスターのアクティブなアプリケーションに適用されます。



注記

ルール **id** は一意である必要があります。この例では、id 0 のクラッシュルールがもう 1 つしかないため、id 1 を使用しています。デプロイメントにさらにルールが作成されている場合は、次の空き ID を使用します。

宣言された CRUSH ルールには、次の情報が含まれています。

- **ルール名**
  - 説明: ルールを識別する一意の完全な名前。
  - 値: **stretch_rule**

- **id**
  - 説明: ルールを識別する一意の整数。
  - 値: **1**
- **type**
  - 説明: レプリケートまたはイレイジャーコーディングされたストレージドライブのルールを説明しています。
  - 値: **replicated**
- **min_size**
  - 説明: プールがこの数よりも小さいレプリカを使用する場合、CRUSH はこのルールを選択しません。
  - 値: 1
- **max_size**
  - 説明: プールがこの数よりも大きいレプリカを使用する場合、CRUSH はこのルールを選択しません。
  - 値: 10
- **step take default**
  - 説明: **default** という名のルートバケットを取得し、ツリーの下方への反復を開始します。
- **step choose firstn 0 type datacenter**
  - 説明: データセンターのバケットを選択し、そのサブツリーに入ります。
- **step chooseleaf firstn 2 type host**
  - 説明: 指定されたタイプのバケットの数を選択します。この場合、前のレベルで入力したデータセンターにある 2 つの異なるホストです。
- **step emit**
  - 説明: 現在の値を出力し、スタックを除算します。通常、ルールの最後に使用されますが、同じルール内の異なるツリーを選択する際に使用することもできます。

10. ファイル **/etc/ceph/crushmap.txt** から新しい CRUSH マップをコンパイルし、これを **/etc/ceph/crushmap2.bin** というバイナリーファイルに変換します。

```
$ crushtool -c /etc/ceph/crushmap.txt -o /etc/ceph/crushmap2.bin
```

11. 作成した新しいクラッシュマップをクラスターに注入します。

```
$ ceph osd setcrushmap -i /etc/ceph/crushmap2.bin
```

出力例:

```
17
```

-

注記

数字の 17 はカウンターであり、クラッシュマップに加えた変更に応じて増加します (18、19 など)。

12. 作成したストレッチルールが使用可能になったことを確認します。

```
ceph osd crush rule ls
```

出力例:

```
replicated_rule
stretch_rule
```

13. ストレッチクラスターモードを有効にします。

```
$ ceph mon enable_stretch_mode ceph7 stretch_rule datacenter
```

この例では、**ceph7** が arbiter ノード、 **stretch_rule** が前の手順で作成したクラッシュルール、**datacenter** が分割バケットです。

14. すべてのプールが、Ceph クラスターに作成した **stretch_rule** CRUSH ルールを使用していることを確認します。

```
$ for pool in $(rados lspools);do echo -n "Pool: ${pool}; ";ceph osd pool get ${pool}
crush_rule;done
```

出力例:

```
Pool: device_health_metrics; crush_rule: stretch_rule
Pool: cephfs.cephfs.meta; crush_rule: stretch_rule
Pool: cephfs.cephfs.data; crush_rule: stretch_rule
Pool: .rgw.root; crush_rule: stretch_rule
Pool: default.rgw.log; crush_rule: stretch_rule
Pool: default.rgw.control; crush_rule: stretch_rule
Pool: default.rgw.meta; crush_rule: stretch_rule
Pool: rbdpool; crush_rule: stretch_rule
```

これは、arbiter モードで稼働中の Red Hat Ceph Storage ストレッチクラスターが利用可能になったことを示しています。

3.6. マネージドクラスターへの OPENSHIFT DATA FOUNDATION のインストール

2 つの OpenShift Container Platform クラスター間でストレージレプリケーションを設定するには、先に OpenShift Data Foundation Operator を各マネージドクラスターにインストールする必要があります。

前提条件

- OpenShift Data Foundation の外部デプロイメントのハードウェア要件を満たしていることを確認してください。ハードウェア要件の詳細は、外部モードの要件 を参照してください。

手順

1. 各管理対象クラスターに最新の **OpenShift Data Foundation** クラスターをインストールして設定します。

2. Operator をインストールした後、オプション **Full デプロイメント** タイプおよび **Connect with external storage platform** を使用して、**バッキングストレージタイプ** が **Red Hat Ceph Storage** である StorageSystem を作成します。
詳しい手順は、外部モードでの OpenShift Data Foundation のデプロイ を参照してください。

   ceph-external-cluster-details-exporter.py スクリプトで次のフラグを使用します。

   a. 少なくとも、**ceph-external-cluster-details-exporter.py script** で次の 3 つのフラグを使用する必要があります。

   --rbd-data-pool-name

   OpenShift Container Platform の RHCS デプロイメント中に作成された RBD プールの名前を使用します。たとえば、プールを **rbdpool** と呼ぶことができます。

   --rgw-endpoint

   <ip_address>:<port> の形式でエンドポイントを指定します。これは、設定している OpenShift Container Platform クラスターと同じサイトで実行されている RGW デーモンの RGW IP です。

   --run-as-user

   サイトごとに異なるクライアント名を使用します。

   b. RHCS の展開中にデフォルト値が使用された場合、次のフラグは **optional** です。

   --cephfs-filesystem-name

   OpenShift Container Platform の RHCS デプロイメント中に作成した CephFS ファイルシステムの名前を使用すると、デフォルトのファイルシステム名は **cephfs** になります。

   --cephfs-data-pool-name

   OpenShift Container Platform の RHCS デプロイメント中に作成した CephFS データプールの名前で、デフォルトプールは **cephfs.data** と呼ばれます。

   --cephfs-metadata-pool-name

   OpenShift Container Platform の RHCS デプロイメント中に作成した CephFS メタデータプールの名前で、デフォルトプールは **cephfs.meta** と呼ばれます。

   c. ブートストラップノード **ceph1** で次のコマンドを実行して、datacenter1 と datacenter2 の RGW エンドポイントの IP を取得します。

   ```
   ceph orch ps | grep rgw.objectgw
   ```

   出力例:

   ```
   rgw.objectgw.ceph3.mecpzm  ceph3  *:8080       running (5d)     31s ago   7w     204M
   -  16.2.7-112.el8cp
   rgw.objectgw.ceph6.mecpzm  ceph6  *:8080       running (5d)     31s ago   7w     204M
   -  16.2.7-112.el8cp
   ```

```
host ceph3.example.com
host ceph6.example.com
```

出力例:

```
ceph3.example.com has address 10.0.40.24
ceph6.example.com has address 10.0.40.66
```

d. ブートストラップされたノード **ceph1** 上の最初の OpenShift Container Platform マネージドクラスター **cluster1** 用に設定されたパラメーターを使用して **ceph-external-cluster-details-exporter.py** を実行します。

```
python3 ceph-external-cluster-details-exporter.py --rbd-data-pool-name rbdpool --cephfs-filesystem-name cephfs --cephfs-data-pool-name cephfs.cephfs.data  --cephfs-metadata-pool-name cephfs.cephfs.meta --<rgw-endpoint> XXX.XXX.XXX.XXX:8080 --run-as-user client.odf.cluster1 > ocp-cluster1.json
```

注記

環境に応じて <rgw-endpoint> XXX.XXX.XXX.XXX を変更します。

e. ブートストラップされたノード **ceph1** 上の最初の OpenShift Container Platform マネージドクラスター **cluster2** 用に設定されたパラメーターを使用して **ceph-external-cluster-details-exporter.py** を実行します。

```
python3 ceph-external-cluster-details-exporter.py --rbd-data-pool-name rbdpool --cephfs-filesystem-name cephfs --cephfs-data-pool-name cephfs.cephfs.data  --cephfs-metadata-pool-name cephfs.cephfs.meta --rgw-endpoint XXX.XXX.XXX.XXX:8080 --run-as-user client.odf.cluster2 > ocp-cluster2.json
```

注記

環境に応じて <rgw-endpoint> XXX.XXX.XXX.XXX を変更します。

- ブートストラップクラスター (ceph1) で生成された 2 つのファイル **ocp-cluster1.json** と **ocp-cluster2.json** をローカルマシンに保存します。
- 外部 OpenShift Data Foundation がデプロイされている **cluster1** 上の OpenShift Container Platform コンソールで、ファイル **ocp-cluster1.json** の内容を使用します。
- 外部 OpenShift Data Foundation がデプロイされている **cluster2** 上の OpenShift Container Platform コンソールで、ファイル **ocp-cluster2.json** の内容を使用します。

3. 設定を確認し、**Create StorageSystem** を選択します。

4. 以下のコマンドを使用して、各管理対象クラスターで OpenShift Data Foundation が正常にデプロイされたことを検証します。

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

Multicloud Gateway (MCG) の場合:

-

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

ステータスの結果が、**プライマリーマネージドクラスター** と **セカンダリーマネージドクラスター** の両方のクエリーに対して Ready になるまで待ちます。

5. OpenShift Web コンソールで、**Installed Operators → OpenShift Data Foundation → Storage System → ocs-external-storagecluster-storagesystem → Resources** の順に移動します。**StorageCluster** の **Status** が **Ready** で、横に緑色のチェックマークが付いていることを確認します。

6. RBD と CephFS ボリュームの読み取りアフィニティーを、最も近いデータセンターから提供できるようにします。

   a. プライマリーマネージドクラスター で、すべてのノードにラベルを付けます。

   ```
   $ oc label nodes --all metro-dr.openshift-storage.topology.io/datacenter=DC1
   ```

   以下のコマンドを実行して、読み取りアフィニティーを有効にします。

   ```
   $ oc patch storageclusters.ocs.openshift.io -n openshift-storage ocs-external-storagecluster -p '{"spec":{"csi":{"readAffinity":{"enabled":true,"crushLocationLabels":["metro-dr.openshift-storage.topology.io/datacenter"]}}}}' --type=merge
   ```

   ```
   $ oc delete po -n openshift-storage -l 'app in (csi-cephfsplugin,csi-rbdplugin)'
   ```

   b. セカンダリーマネージドクラスターで、すべてのノードにラベルを付けます。

   ```
   $ oc label nodes --all metro-dr.openshift-storage.topology.io/datacenter=DC2
   ```

   以下のコマンドを実行して、読み取りアフィニティーを有効にします。

   ```
   $ oc patch storageclusters.ocs.openshift.io -n openshift-storage ocs-external-storagecluster -p '{"spec":{"csi":{"readAffinity":{"enabled":true,"crushLocationLabels":["metro-dr.openshift-storage.topology.io/datacenter"]}}}}' --type=merge
   ```

   ```
   $ oc delete po -n openshift-storage -l 'app in (csi-cephfsplugin,csi-rbdplugin)'
   ```

3.7. OPENSHIFT DATA FOUNDATION MULTICLUSTER ORCHESTRATOR OPERATOR のインストール

OpenShift Data Foundation のマルチクラスターオーケストレーターは、ハブクラスターの OpenShift Container Platform の OperatorHub からインストールされるコントローラーです。

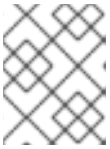
手順

1. **ハブクラスター** で **OperatorHub** に移動し、キーワードフィルターを使用して **ODF Multicluster Orchestrator** を検索します。

2. **ODF Multicluster Orchestrator** タイルをクリックします。

3. すべてのデフォルト設定をそのままにして、**Install** をクリックします。

Operator のリソースが **openshift-operators** プロジェクトにインストールされ、すべての namespace で利用できることを確認してください。



注記

ODF Multicluster Orchestrator レーターは、依存関係として RHACM ハブクラスターに **Openshift DR ハブ Operator** もインストールします。

4. Operator **Pod** が **Running** 状態であることを確認します。**OpenShift DR Hub Operator** も同時に **openshift-operators** namespace にインストールされます。

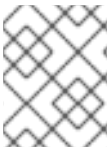
```
$ oc get pods -n openshift-operators
```

出力例:

```
NAME                                  READY   STATUS      RESTARTS    AGE
odf-multicluster-console-6845b795b9-blxrn   1/1     Running     0           4d20h
odfmo-controller-manager-f9d9dfb59-jbrsd    1/1     Running     0           4d20h
ramen-hub-operator-6fb887f885-fss4w         2/2     Running     0           4d20h
```

3.8. クラスター全体での SSL アクセスの設定

プライマリークラスターとセカンダリークラスター 間のネットワーク (SSL) アクセスを設定して、メタデータを代替クラスターのマルチクラウドゲートウェイ (MCG) **object bucket** に安全なトランスポートプロトコルを使用して格納し、**ハブクラスター** に格納してオブジェクトバケットへのアクセスを検証できるようにします。



注記

すべての OpenShift クラスターが環境の署名済みの有効な証明書セットを使用してデプロイされる場合は、このセクションを省略できます。

手順

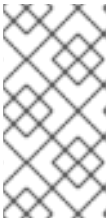
1. プライマリーマネージドクラスター の Ingress 証明書をデプロイメントし、出力を **primary.crt** に保存します。

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

2. セカンダリーマネージドクラスターの Ingress 証明書を抽出し、出力を **secondary.crt** に保存します。

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

3. リモートクラスターの証明書バンドルを保持する新しい **ConfigMap** ファイルを、ファイル名 **cm-clusters-crt.yaml** で作成します。



注記

この例のように、クラスターごとに 3 つ以上の証明書が存在する可能性があります。また、以前作成した **primary.crt** ファイルおよび **secondary.crt** ファイルから、証明書の内容をコピーして貼り付けた後に、証明書の内容が正しくインデントされていることを確認します。

```
apiVersion: v1
data:
  ca-bundle.crt: |
    -----BEGIN CERTIFICATE-----
    <copy contents of cert1 from primary.crt here>
    -----END CERTIFICATE-----

    -----BEGIN CERTIFICATE-----
    <copy contents of cert2 from primary.crt here>
    -----END CERTIFICATE-----

    -----BEGIN CERTIFICATE-----
    <copy contents of cert3 primary.crt here>
    -----END CERTIFICATE----

    -----BEGIN CERTIFICATE-----
    <copy contents of cert1 from secondary.crt here>
    -----END CERTIFICATE-----

    -----BEGIN CERTIFICATE-----
    <copy contents of cert2 from secondary.crt here>
    -----END CERTIFICATE-----

    -----BEGIN CERTIFICATE-----
    <copy contents of cert3 from secondary.crt here>
    -----END CERTIFICATE-----
kind: ConfigMap
metadata:
  name: user-ca-bundle
  namespace: openshift-config
```

4. **プライマリーマネージドクラスター**、**セカンダリーマネージドクラスター**、**ハブクラスター** で **ConfigMap** を作成します。

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

出力例:

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

5. **プライマリーマネージドクラスター**、**セカンダリーマネージドクラスター**、および **ハブクラスター** 上のデフォルトのプロキシーリソースにパッチを適用します。

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

出力例:

```
proxy.config.openshift.io/cluster patched
```

3.9. ハブクラスターでの障害復旧ポリシーの作成

Openshift 災害復旧ポリシー (DRPolicy) リソースは、災害復旧ソリューションに参加する OpenShift Container Platform クラスターと目的のレプリケーション間隔を指定します。DRPolicy は、ユーザーが障害復旧ソリューションを必要とするアプリケーションに適用できるクラスタースコープのリソースです。

ODF MultiCluster Orchestrator Operator は、**Multicluster Web コンソール** を介して、各 **DRPolicy** および対応する **DRClusters** の作成を容易にします。

前提条件

- 2 つのマネージドクラスターの最小セットがあることを確認します。

手順

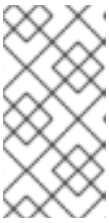
1. **OpenShift コンソール** で、**All Clusters → Data Services → Data policies** に移動します。
2. **DR ポリシーの作成** をクリックします。
3. **ポリシー名** を入力します。各 DRPolicy に一意の名前が付けられていることを確認します (例: **ocp4perf1-ocp4perf2**)。
4. この新しいポリシーが関連付けられる管理対象クラスターのリストから 2 つのクラスターを選択します。
5. **レプリケーションポリシー** は、選択した OpenShift クラスターに基づいて **sync** するように自動的に設定されます。
6. **Create** をクリックします。
7. **DRPolicy** が正常に作成されたことを確認します。作成された各 DRPolicy リソースごとに **ハブクラスター** でこのコマンドを実行します。**<drpolicy_name>** は、一意の名前に置き換えてください。

   ```
   $ oc get drpolicy <drpolicy_name> -o jsonpath='{.status.conditions[].reason}{"\n"}'
   ```

   出力例:

   ```
   Succeeded
   ```

   DRPolicy が作成されると、それに伴って 2 つの DRCluster リソースも作成されます。3 つのリソースすべてが検証され、ステータスが **Succeeded** と表示されるまで、最大 10 分かかる場合があります。



注記

DRPolicy で
は、**SchedulingInterval**、**ReplicationClassSelector**、**VolumeSnapshotClass Selector**、および **DRClusters** フィールドの値の編集はサポートされていません。

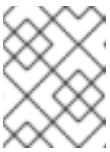
8. **ハブクラスター** から **プライマリーマネージドクラスター** と **セカンダリーマネージドクラスター** の両方へのオブジェクトバケットアクセスを確認します。

   a. ハブクラスター上の **DRClusters** の名前を取得します。

   ```
   $ oc get drclusters
   ```

   出力例:

   b. 各マネージドクラスター上に作成された各バケットへの S3 アクセスを確認します。DRCluster 検証コマンドを使用します。**<drcluster_name>** は一意の名前に置き換えてください。

   注記

   DRCluster では、**Region** および **S3ProfileName** フィールド値の編集はサポートされていません。

   ```
   $ oc get drcluster <drcluster_name> -o jsonpath='{.status.conditions[2].reason}{"\n"}'
   ```

   出力例:

   ```
   Succeeded
   ```

   注記

   Hub cluster の両方の **DRClusters** に対してコマンドを実行してください。

9. **OpenShift DR Cluster Operator** のインストールが **プライマリー管理対象クラスター** および **セカンダリー管理対象** クラスターで成功したことを確認します。

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

   出力例:

   ```
   NAME                                                              DISPLAY                       VERSION
   REPLACES   PHASE
   clusterserviceversion.operators.coreos.com/odr-cluster-operator.v4.15.0         Openshift DR
   Cluster Operator   4.15.0                    Succeeded
   clusterserviceversion.operators.coreos.com/volsync-product.v0.8.0               VolSync
   0.8.0                     Succeeded

   NAME                                      READY   STATUS    RESTARTS   AGE
   pod/ramen-dr-cluster-operator-6467cf5d4c-cc8kz   2/2     Running   0          3d12h
   ```

   各管理対象クラスターの **OperatorHub** に **OpenShift DR Cluster Operator** が正常にインストールされていることを確認することもできます。

10. シークレットがプライマリーマネージドクラスターとセカンダリーマネージドクラスターに正しく伝播されていることを確認します。

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

出力をハブクラスターからの s3SecretRef と照合します。

```
oc get cm -n openshift-operators ramen-hub-operator-config -oyaml
```

3.10. フェンシングの自動化のために DRCLUSTERS を設定する

この設定は、アプリケーションのフェイルオーバーの前にフェンシングを有効にするために必要です。災害に見舞われたクラスターから永続ボリュームへの書き込みを防ぐために、OpenShift DR は Red Hat Ceph Storage (RHCS) に、クラスターのノードを RHCS 外部ストレージからフェンシングするように指示します。このセクションでは、DRCluster のノードに IP または IP 範囲を追加する方法について説明します。

3.10.1. ノード IP アドレスを DRClusters に追加する

1. **プライマリーマネージドクラスター** および **セカンダリーマネージドクラスター** でこのコマンドを実行して、マネージドクラスター内のすべての OpenShift ノードの **IP アドレス** を見つけます。

```
$ oc get nodes -o jsonpath='{range .items[*]}{.status.addresses[?
(@.type=="ExternalIP")].address}{"\n"}{end}'
```

出力例:

```
10.70.56.118
10.70.56.193
10.70.56.154
10.70.56.242
10.70.56.136
10.70.56.99
```

IP addresses を取得したら、管理対象クラスターごとに **DRCluster** リソースを変更できます。

2. ハブクラスターで **DRCluster** 名を見つけます。

```
$ oc get drcluster
```

出力例:

```
NAME        AGE
ocp4perf1   5m35s
ocp4perf2   5m35s
```

3. **<drcluster_name>** を一意の名前に置き換えた後、各 **DRCluster** を編集して一意の IP アドレスを追加します。

```
$ oc edit drcluster <drcluster_name>
```

```
apiVersion: ramendr.openshift.io/v1alpha1
kind: DRCluster
metadata:
[...]
```

```
spec:
  s3ProfileName: s3profile-<drcluster_name>-ocs-external-storagecluster
  ## Add this section
  cidrs:
    - <IP_Address1>/32
    - <IP_Address2>/32
    - <IP_Address3>/32
    - <IP_Address4>/32
    - <IP_Address5>/32
    - <IP_Address6>/32
[...]
```

出力例:

```
drcluster.ramendr.openshift.io/ocp4perf1 edited
```

注記

6 つを超える IP アドレスが存在する可能性があります。

ピア DRCluster リソース (ocp4perf2 など) の **セカンダリー管理クラスター** の **IP addresses** についても、この **DRCluster** 設定を変更します。

3.10.2. DRClusters にフェンシングアノテーションを追加する

次の注釈をすべての DRCluster リソースに追加します。これらの注釈には、これらの手順の後半で作成される **NetworkFence** リソースに必要な詳細が含まれています (アプリケーションのフェイルオーバーをテストする前に)。

注記

<drcluster_name> を一意の名前に置き換えます。

```
$ oc edit drcluster <drcluster_name>
```

```
apiVersion: ramendr.openshift.io/v1alpha1
kind: DRCluster
metadata:
  ## Add this section
  annotations:
    drcluster.ramendr.openshift.io/storage-clusterid: openshift-storage
    drcluster.ramendr.openshift.io/storage-driver: openshift-storage.rbd.csi.ceph.com
    drcluster.ramendr.openshift.io/storage-secret-name: rook-csi-rbd-provisioner
    drcluster.ramendr.openshift.io/storage-secret-namespace: openshift-storage
[...]
```

出力例:

```
drcluster.ramendr.openshift.io/ocp4perf1 edited
```

両方の **DRCluster** リソースにこれらのアノテーションを必ず追加してください (例: **ocp4perf1** と **ocp4perf2**)。

3.11. 障害復旧ソリューションをテストするためのサンプルアプリケーションを作成する

OpenShift Data Foundation 障害復旧 (DR) ソリューションは、RHACM によって管理されるサブスクリプションベースおよびアプリケーションセットベースのアプリケーションの障害復旧をサポートします。詳細は、サブスクリプション と ApplicationSet のドキュメントを参照してください。

次のセクションでは、アプリケーションを作成して DRPolicy をアプリケーションに適用する方法について詳しく説明します。

- サブスクリプションベースのアプリケーション
  cluster-admin パーミッションを持たない OpenShift ユーザーは、ナレッジ記事 で、障害復旧アクションを実行するために必要なパーミッションをアプリケーションユーザーに割り当てる方法を参照してください。

- ApplicationSet ベースのアプリケーション
  クラスター管理者権限を持たない OpenShift ユーザーは、ApplicationSet ベースのアプリケーションを作成できません。

3.11.1. サブスクリプションベースのアプリケーション

3.11.1.1. サンプルのサブスクリプションベースのアプリケーションを作成する

プライマリーマネージドクラスター から **セカンダリーマネージドクラスター** への **failover** および **relocate** をテストするには、サンプルアプリケーションが必要です。

前提条件

- 一般消費用のアプリケーションを作成する場合は、アプリケーションが 1 つのクラスターのみにデプロイされるようにします。
- **busybox** というサンプルアプリケーションを例として使用します。
- アプリケーションのすべての外部ルートが、アプリケーションがフェイルオーバーまたは再配置されたときのトラフィックリダイレクト用に Global Traffic Manager (GTM) または Global Server Load Balancing (GLSB) サービスを使用して設定されていることを確認します。
- ベストプラクティスとして、一緒に属する Red Hat Advanced Cluster Management (RHACM) サブスクリプションをグループ化し、それらをグループとして DR で保護する単一の配置ルールを参照します。さらに、フェイルオーバーや再配置などの将来の DR アクションのために、その配置ルールを、サブスクリプションを論理的にグループ化する単一のアプリケーションとして作成します。

注記

関連のないサブスクリプションが配置アクションで同じ配置ルールを参照している場合、配置ルールを参照するすべてのサブスクリプションが DR ワークフローによって制御されるため、これらのサブスクリプションも DR で保護されます。

手順

1. ハブクラスターで、**Applications** に移動し、**Create application** をクリックします。
2. 種類は **Subscription**を選択します。

3. アプリケーションの Name (**busybox** など) および Namespace (**busybox-sample** など) を入力します。

4. Repository location for resources セクションで **Repository type Git** を選択します。

5. サンプルアプリケーションの github **Branch** および **Path** で、 Git リポジトリー URL を入力します。リソース **busybox** Pod および PVC が作成されます。
サンプルアプリケーションリポジトリーを **https://github.com/red-hat-storage/ocm-ramen-samples** として使用します。**Branch** は **release-4.15**、**Path** は **busybox-odr-metro** に置き換えます。

6. **Deploy application resources on clusters with all specified labels**が表示されるまで、フォームを下にスクロールします。

   - グローバル **Cluster sets**、またはお使いの環境で、正しいマネージドクラスターが含まれるクラスターを選択します。

   - ラベル **<name>** を追加し、その値を **マネージドクラスター** 名に設定します。

7. 右上隅にある **作成** をクリックします。
後続の画面で、**Topology** タブに移動します。アプリケーショントポロジーのチェックマークがすべて緑であることが確認できるはずです。

注記

詳細な情報を表示するには、トポロジー要素のいずれかをクリックすると、トポロジービューの右側にウィンドウが表示されます。

8. サンプルアプリケーションのデプロイを検証しています。
busybox アプリケーションが優先クラスターにデプロイされたので、デプロイを検証できます。

   RHACM によって **busybox** がデプロイされたマネージドクラスターにログインします。

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

   出力例:

```
NAME                      READY   STATUS    RESTARTS   AGE
pod/busybox-67bf494b9-zl5tr   1/1     Running   0          77s


NAME                         STATUS   VOLUME                                     CAPACITY   ACCESS
MODES   STORAGECLASS            AGE
persistentvolumeclaim/busybox-pvc   Bound    pvc-c732e5fe-daaf-4c4d-99dd-462e04c18412
5Gi        RWO            ocs-storagecluster-ceph-rbd   77s
```

3.11.1.2. サンプルアプリケーションにデータポリシーを適用する

前提条件

- Data ポリシーで参照されている両方の管理対象クラスターが到達可能であることを確認する。そうでない場合、両方のクラスターがオンラインになるまで、アプリケーションは災害復旧で保護されません。

手順

1. ハブクラスターで、**All Clusters** → **Applications** に移動します。
2. アプリケーションの最後にあるアクションメニューをクリックして、使用可能なアクションのリストを表示します。
3. **Manage data policy**→ **Assign data policy**をクリックします。
4. **Policy** を選択し、**Next** をクリックします。
5. **Application resource** を選択し、**PVC label selector** を使用して、選択したアプリケーションリソースの **PVC label** を選択します。

注記

選択したアプリケーションリソースに対して複数の PVC ラベルを選択できます。**Add application resource**オプションを使用して、複数のリソースを追加することもできます。

6. すべてのアプリケーションリソースを追加したら、**Next** をクリックします。
7. **Policy configuration details** を確認し、**Assign** をクリックします。新しく割り当てられたデータポリシーが **Manage data policy** モーダルリストビューに表示されます。
8. アプリケーションページで、割り当てられたポリシーの詳細を表示できることを確認します。
    a. アプリケーションページで、**Data policy** 列に移動し、**policy link** をクリックしてビューをデプロイメントします。
    b. 割り当てられたポリシーの数と、フェイルオーバーおよび再配置のステータスが表示されることを確認します。
    c. **View more details** をクリックして、アプリケーションで使用されているポリシーによる進行中のアクティビティーのステータスを表示します。
9. DRPolicy をアプリケーションに適用した後、drpc yaml 出力で **ClusterDataProtected** が **True** に設定されているかどうかを確認します。

3.11.2. ApplicationSet ベースのアプリケーション

3.11.2.1. ApplicationSet ベースのアプリケーションの作成

前提条件

- Red Hat OpenShift GitOps Operator が Hub クラスターにインストールされている。手順は、RHACM のドキュメント を参照してください。
- プライマリーとセカンダリーの両方のマネージドクラスターが GitOps に登録されていることを確認します。登録手順は、GitOps へのマネージドクラスターの登録 を参照してください。次に、両方のマネージドクラスターを登録するために **GitOpsCluster** リソースによって使用される配置に、クラスターが使用できない場合に対処するための許容機能があるかどうかを確認します。コマンド **oc get placement-name> -n openshift-gitops -o yaml** を使用して、次の許容範囲が Placement に追加されているかどうかを確認できます。

```
tolerations:
- key: cluster.open-cluster-management.io/unreachable
  operator: Exists
- key: cluster.open-cluster-management.io/unavailable
  operator: Exists
```

許容が追加されない場合は、 Red Hat Advanced Cluster Management および OpenShift GitOps のアプリケーション配置許容範囲の設定 を参照してください。

手順

1. ハブクラスターで、**All Clusters** → **Applications** に移動し、**Create application** をクリックします。
2. アプリケーションタイプを **Argo CD ApplicationSet - プッシュモデル** として選択します
3. 一般ステップ 1 で、**Application set name** を入力します。
4. **Argo サーバーopenshift-gitops** を選択し、**Requeue time** を **180** 秒に設定します。
5. **Next** をクリックします。
6. Repository location for resources セクションで **Repository type Git** を選択します。
7. サンプルアプリケーションの github Branch および Path で、 Git リポジトリー URL を入力します。リソース busybox Pod および PVC が作成されます。
   a. サンプルアプリケーションリポジトリーを https://github.com/red-hat-storage/ocm-ramen-samples として使用します。
   b. **Revision** を **release-4.15** として選択します。
   c. パスを **busybox-odr-metro** として選択します。
8. **Remote namespace** の値 (例: busybox-sample) を入力し、 **Next** をクリックします。
9. **Sync policy** を選択し、**Next** をクリックします。
   1 つ以上のオプションを選択できます。
10. ラベル **<name>** を追加し、その値を **マネージドクラスター** 名に設定します。
11. **Next** をクリックします。
12. 設定の詳細を確認し、**Submit** をクリックします。

3.11.2.2. ApplicationSet ベースのサンプルアプリケーションにデータポリシーを適用する

前提条件

- Data ポリシーで参照されている両方の管理対象クラスターが到達可能であることを確認する。そうでない場合、両方のクラスターがオンラインになるまで、アプリケーションは災害復旧で保護されません。

手順

1. ハブクラスターで、**All Clusters** → **Applications** に移動します。

2. アプリケーションの最後にあるアクションメニューをクリックして、使用可能なアクションのリストを表示します。
3. **Manage data policy→ Assign data policy**をクリックします。
4. **Policy** を選択し、**Next** をクリックします。
5. **Application resource** を選択し、**PVC label selector** を使用して、選択したアプリケーションリソースの **PVC label** を選択します。

注記

選択したアプリケーションリソースに対して複数の PVC ラベルを選択できます。

6. すべてのアプリケーションリソースを追加したら、**Next** をクリックします。
7. **Policy configuration details** を確認し、**Assign** をクリックします。新しく割り当てられたデータポリシーが **Manage data policy** モーダルリストビューに表示されます。
8. アプリケーションページで、割り当てられたポリシーの詳細を表示できることを確認します。
   a. アプリケーションページで、**Data policy** 列に移動し、**policy link** をクリックしてビューをデプロイメントします。
   b. 割り当てられたポリシーの数と、フェイルオーバーおよび再配置のステータスが表示されることを確認します。
9. DRPolicy をアプリケーションに適用した後、drpc yaml 出力で **ClusterDataProtected** が **True** に設定されているかどうかを確認します。

3.11.3. サンプルアプリケーションの削除

このセクションでは、RHACM コンソールを使用してサンプルアプリケーションの **Busybox** を削除する手順を説明します。

重要

DR で保護されたアプリケーションを削除する場合は、DRPolicy に属する両方のクラスターへのアクセスが必要です。これは、保護されたすべての API リソースと、それぞれの S3 ストア内のリソースが、DR 保護の削除の一環として確実に消去されるようにするためです。いずれかのクラスターへのアクセスが正常でない場合、ハブ上のアプリケーションの **DRPlacementControl** リソースを削除すると、Deleting の状態のままになります。

前提条件

- サンプルアプリケーションを削除する手順は、フェイルオーバーと再配置のテストが完了し、アプリケーションを RHACM とマネージドクラスターから削除する準備ができるまで実行しないでください。

手順

1. RHACM コンソールで、**Applications** に移動します。

2. 削除するサンプルアプリケーションを検索します (例: **busybox**)。

3. 削除するアプリケーションの横にある Action メニュー (⋮) をクリックします。

4. **Delete application** をクリックします。
Delete application を選択すると、アプリケーション関連のリソースも削除すべきかどうかを求める新規画面が表示されます。

5. **Remove application related resources**チェックボックスを選択して、Subscription および PlacementRule を削除します。

6. **Delete** をクリックします。これにより、Primary マネージドクラスター (またはアプリケーションが実行しているクラスター) の busybox アプリケーションが削除されます。

7. RHACM コンソールを使用して削除されたリソースに加えて、**busybox** アプリケーションの削除後に **DRPlacementControl** が自動削除されない場合は、DRPlacementControl も削除します。

    a. ハブクラスターの OpenShift Web コンソールにログインし、プロジェクト **busybox-sample** の Installed Operators に移動します。
    ApplicationSet アプリケーションの場合は、プロジェクトを **openshift-gitops** として選択します。

    b. **OpenShift DR Hub Operator** をクリックした後、**DRPlacementControl** タブをクリックします。

    c. 削除する **busybox** アプリケーション DRPlacementControl の横にあるアクションメニュー (⋮) をクリックします。

    d. **Delete DRPlacementControl** をクリックします。

    e. **Delete** をクリックします。

注記

このプロセスを使用して、**DRPlacementControl** リソースでアプリケーションを削除できます。

3.12. マネージドクラスター間のサブスクリプションベースのアプリケーションフェイルオーバー

何らかの理由で管理対象クラスターが使用できなくなったときに、フェイルオーバーを実行します。このフェイルオーバー方式はアプリケーションベースです。

前提条件

- セットアップにアクティブおよびパッシブの RHACM ハブクラスターがある場合は、Red Hat Advanced Cluster Management を使用したハブのリカバリー を参照してください。
- プライマリークラスターが **Ready** 以外の状態にある場合は、更新に時間がかかる可能性があるため、クラスターの実際のステータスを確認してください。

    1. **RHACM console → Infrastructure → Clusters → Cluster list**タブに移動します。

    2. フェイルオーバー操作を実行する前に、両方のマネージドクラスターのステータスを個別に確認してください。

ただし、フェイルオーバー先のクラスターが **Ready** 状態にある場合でも、フェイルオーバー操作は実行できます。

手順

1. **Hub cluster** でフェンシングを有効にします。

   a. CLI ターミナルを開き、**DRCluster resource** を編集します。**<drcluster_name>** は一意の名前に置き換えます。

      注意

      管理対象クラスターがフェンシングされると、アプリケーションから OpenShift Data Foundation 外部ストレージクラスターへの **すべての** 通信が失敗し、現在フェンシングされているクラスターで一部の Pod が **異常** な状態になります (例: **CreateContainerError**、CrashLoopBackOff)。

      ```
      $ oc edit drcluster <drcluster_name>
      ```

      ```
      apiVersion: ramendr.openshift.io/v1alpha1
      kind: DRCluster
      metadata:
      [...]
      spec:
        ## Add this line
        clusterFence: Fenced
        cidrs:
        [...]
      [...]
      ```

      出力例:

      ```
      drcluster.ramendr.openshift.io/ocp4perf1 edited
      ```

   b. **Hub cluster** 上の **プライマリーマネージドクラスター** のフェンシングステータスを確認します。<drcluster_name> は、一意の識別子に置き換えます。

      ```
      $ oc get drcluster.ramendr.openshift.io <drcluster_name> -o jsonpath='{.status.phase}
      {"\n"}'
      ```

      出力例:

      ```
      Fenced
      ```

   c. OpenShift Container Platform クラスターノードに属する IP がブロックリストにあることを確認します。

      ```
      $ ceph osd blocklist ls
      ```

      出力例

      ```
      cidr:10.1.161.1:0/32 2028-10-30T22:30:03.585634+0000
      ```

```
cidr:10.1.161.14:0/32 2028-10-30T22:30:02.483561+0000
cidr:10.1.161.51:0/32 2028-10-30T22:30:01.272267+0000
cidr:10.1.161.63:0/32 2028-10-30T22:30:05.099655+0000
cidr:10.1.161.129:0/32 2028-10-30T22:29:58.335390+0000
cidr:10.1.161.130:0/32 2028-10-30T22:29:59.861518+0000
```

2. ハブクラスターで、**Applications** に移動します。
3. アプリケーション行の最後にある **Actions** メニューをクリックして、使用可能なアクションのリストを表示します。
4. **Failover application** をクリックします。
5. **Failover application** モーダルが表示されたら、障害時に関連付けられたアプリケーションがフェイルオーバーする **ポリシー** と **ターゲットクラスター** を選択します。
6. **Select subscription group** ドロップダウンをクリックして、デフォルトの選択を確認するか、この設定を変更します。
   デフォルトでは、アプリケーションリソースをレプリケートするサブスクリプショングループが選択されています。
7. **フェイルオーバーの準備** 状況を確認します。
   - ステータスが **Ready** で緑色のチェックマークが付いている場合は、ターゲットクラスターでフェイルオーバーを開始する準備ができていることを示しています。手順 7 に進みます。
   - ステータスが **Unknown** または **Not ready** の場合は、ステータスが **Ready** に変わるまで待ちます。
8. **Initiate** をクリックします。busybox アプリケーションが現在、**セカンダリーで管理されているクラスター** にフェイルオーバーしています。
9. モーダルウィンドウを閉じ、Applications ページの **Data policy** 列を使用してステータスを追跡します。
10. アプリケーションのアクティビティーステータスが **FailedOver** と表示されていることを確認します。
    a. **Applications → Overview** タブに移動します。
    b. **Data policy** 列で、ポリシーを適用したアプリケーションの **policy** リンクをクリックします。
    c. **Data policy** ポップオーバーで、**View more details** リンクをクリックします。

3.13. マネージドクラスター間の APPLICATIONSET ベースのアプリケーションフェイルオーバー

何らかの理由で管理対象クラスターが使用できなくなったときに、フェイルオーバーを実行します。このフェイルオーバー方式はアプリケーションベースです。

前提条件

- セットアップにアクティブおよびパッシブの RHACM ハブクラスターがある場合は、Red Hat Advanced Cluster Management を使用したハブのリカバリー を参照してください。

- プライマリークラスターが **Ready** 以外の状態にある場合は、更新に時間がかかる可能性があるため、クラスターの実際のステータスを確認してください。

  1. **RHACM console → Infrastructure → Clusters → Cluster list**タブに移動します。

  2. フェイルオーバー操作を実行する前に、両方のマネージドクラスターのステータスを個別に確認してください。
     ただし、フェイルオーバー先のクラスターが **Ready** 状態にある場合でも、フェイルオーバー操作は実行できます。

手順

1. **Hub cluster** でフェンシングを有効にします。

   a. CLI ターミナルを開き、**DRCluster resource** を編集します。**<drcluster_name>** は一意の名前に置き換えます。

      注意

      管理対象クラスターがフェンシングされると、アプリケーションから OpenShift Data Foundation 外部ストレージクラスターへの **すべての** 通信が失敗し、現在フェンシングされているクラスターで一部の Pod が **異常** な状態になります (例: **CreateContainerError**、**CrashLoopBackOff**)。

      ```
      $ oc edit drcluster <drcluster_name>
      ```

      ```
      apiVersion: ramendr.openshift.io/v1alpha1
      kind: DRCluster
      metadata:
      [...]
      spec:
        ## Add this line
        clusterFence: Fenced
        cidrs:
        [...]
      [...]
      ```

      出力例:

      ```
      drcluster.ramendr.openshift.io/ocp4perf1 edited
      ```

   b. **Hub cluster** 上の **プライマリーマネージドクラスター** のフェンシングステータスを確認します。<drcluster_name> は、一意の識別子に置き換えます。

      ```
      $ oc get drcluster.ramendr.openshift.io <drcluster_name> -o jsonpath='{.status.phase}
      {"\n"}'
      ```

      出力例:

      ```
      Fenced
      ```

   c. OpenShift Container Platform クラスターノードに属する IP がブロックリストにあることを確認します。

```
$ ceph osd blocklist ls
```

出力例

```
cidr:10.1.161.1:0/32 2028-10-30T22:30:03.585634+0000
cidr:10.1.161.14:0/32 2028-10-30T22:30:02.483561+0000
cidr:10.1.161.51:0/32 2028-10-30T22:30:01.272267+0000
cidr:10.1.161.63:0/32 2028-10-30T22:30:05.099655+0000
cidr:10.1.161.129:0/32 2028-10-30T22:29:58.335390+0000
cidr:10.1.161.130:0/32 2028-10-30T22:29:59.861518+0000
```

2. ハブクラスターで、**Applications** に移動します。
3. アプリケーション行の最後にある **Actions** メニューをクリックして、使用可能なアクションのリストを表示します。
4. **Failover application** をクリックします。
5. **フェイルオーバーアプリケーション** モーダルが表示されたら、表示された詳細が正しいことを確認し、フェイルオーバーの **準備状況** のステータスを確認します。ステータスが **Ready** で緑色のチェックマークが付いている場合は、ターゲットクラスターでフェイルオーバーを開始する準備ができていることを示しています。
6. **Initiate** をクリックします。busybox リソースがターゲットクラスターに作成されました。
7. モーダルウィンドウを閉じ、Applications ページの **Data policy** 列を使用してステータスを追跡します。
8. アプリケーションのアクティビティーステータスが **FailedOver** と表示されていることを確認します。
   a. **Applications → Overview** タブに移動します。
   b. **Data policy** 列で、ポリシーを適用したアプリケーションの **policy** リンクをクリックします。
   c. **Data policy** ポップオーバーで、1 つ以上のポリシー名と、アプリケーションで使用されているポリシーに関連付けられた進行中のアクティビティーが表示されていることを確認します。

3.14. マネージドクラスター間でのサブスクリプションベースのアプリケーションの再配置

すべてのマネージドクラスターが使用可能になったら、アプリケーションを適切な場所に再配置します。

前提条件

- セットアップにアクティブおよびパッシブの RHACM ハブクラスターがある場合は、Red Hat Advanced Cluster Management を使用したハブのリカバリー を参照してください。
- プライマリークラスターが **Ready** 以外の状態にある場合は、更新に時間がかかる可能性があるため、クラスターの実際のステータスを確認してください。再配置は、プライマリークラスターと優先クラスターの両方が稼働している場合にのみ実行できます。

1. **RHACM console → Infrastructure → Clusters → Cluster list**タブに移動します。
2. 再配置操作を実行する前に、両方のマネージドクラスターのステータスを個別に確認してください。

- フェンシングを解除する前に、アプリケーションがクラスターからクリーンアップされたことを確認してください。

手順

1. ハブクラスターでフェンシングを無効にします。
   a. このクラスターの **DRCluster リソース** を編集し、<drcluster_name> を一意の名前に置き換えます。

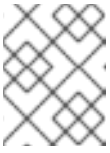
```
$ oc edit drcluster <drcluster_name>
```

```
apiVersion: ramendr.openshift.io/v1alpha1
kind: DRCluster
metadata:
[...]
spec:
  cidrs:
  [...]
  ## Modify this line
  clusterFence: Unfenced
  [...]
[...]
```

   出力例:

```
drcluster.ramendr.openshift.io/ocp4perf1 edited
```

   b. **Fenced** であった OpenShift Container Platform ノードを正常に再起動します。リカバリーオーケストレーションの障害がさらに発生しないように、フェンシング解除後に I/O 操作を再開するには、再起動が必要です。ノードの正常な再起動 の手順に従って、クラスターのすべてのノードを再起動します。



注記

ノードで再起動して uncordon 操作を実行する前に、すべてのノードが最初に接続解除され、ドレインされていることを確認してください。

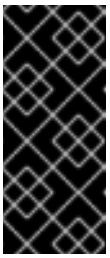
   c. すべての OpenShift ノードが再起動され、**Ready** ステータスになったら、プライマリーマネージドクラスター (または Unfenced されたクラスター) でこのコマンドを実行して、すべての Pod が正常な状態であることを確認します。

```
oc get pods -A | egrep -v 'Running|Completed'
```

   出力例:

```
NAMESPACE                              NAME
READY   STATUS      RESTARTS      AGE
```

次のステップに進む前に、このクエリーの出力は 0 Pod である必要があります。



重要

ストレージ通信が切断されたために Pod がまだ異常な状態にある場合は、続行する前にトラブルシューティングを行って解決してください。ストレージクラスターは OpenShift の外部にあるため、OpenShift アプリケーションを正常に動作させるには、サイトの停止後にストレージクラスターを適切に復元する必要もあります。

または、OpenShift Web コンソールのダッシュボードと 概要 タブを使用して、アプリケーションと外部 ODF ストレージクラスターの正常性を評価することもできます。OpenShiftDataFoundation ダッシュボードの詳細は、**Storage → Data Foundation**に移動すると表示されます。

d. **Unfenced** クラスターが正常な状態であることを確認します。<drcluster_name> は、一意の名前に置き換えて、プライマリー管理クラスターのハブクラスターのフェンシングステータスを検証します。

```
$ oc get drcluster.ramendr.openshift.io <drcluster_name> -o jsonpath='{.status.phase}
{"\n"}'
```

出力例:

```
Unfenced
```

e. OpenShift Container Platform クラスターノードに属する IP がブロックリストに含まれていないことを確認します。

```
$ ceph osd blocklist ls
```

フェンシング中に追加された IP が表示されていないことを確認します。

2. ハブクラスターで、**Applications** に移動します。

3. アプリケーション行の最後にある **Actions** メニューをクリックして、使用可能なアクションのリストを表示します。

4. **Relocate application** をクリックします。

5. **Relocate application** モーダルが表示されたら、障害時に関連付けられたアプリケーションを再配置する **ポリシー** と **ターゲットクラスター** を選択します。

6. デフォルトでは、アプリケーションリソースをデプロイするサブスクリプショングループが選択されています。**Select subscription group** ドロップダウンをクリックして、デフォルトの選択を確認するか、この設定を変更します。

7. **再配置の準備** 状況を確認します。

   - ステータスが **Ready** で緑色のチェックマークが付いている場合は、ターゲットクラスターで再配置を開始する準備ができていることを示しています。手順 7 に進みます。

   - ステータスが **Unknown** または **Not ready** の場合は、ステータスが **Ready** に変わるまで待ちます。

8. **Initiate** をクリックします。busybox リソースがターゲットクラスターに作成されました。
9. モーダルウィンドウを閉じ、Applications ページの **Data policy** 列を使用してステータスを追跡します。
10. アプリケーションのアクティビティーステータスが **Relocated** と表示されていることを確認します。
    a. **Applications → Overview** タブに移動します。
    b. **Data policy** 列で、ポリシーを適用したアプリケーションの **policy** リンクをクリックします。
    c. **Data policy** ポップオーバーで、**View more details** リンクをクリックします。

3.15. APPLICATIONSET ベースアプリケーションのマネージドクラスター間での再配置

すべてのマネージドクラスターが使用可能になったら、アプリケーションを適切な場所に再配置します。

前提条件

- セットアップにアクティブおよびパッシブの RHACM ハブクラスターがある場合は、Red Hat Advanced Cluster Management を使用したハブのリカバリー を参照してください。
- プライマリークラスターが **Ready** 以外の状態にある場合は、更新に時間がかかる可能性があるため、クラスターの実際のステータスを確認してください。再配置は、プライマリークラスターと優先クラスターの両方が稼働している場合にのみ実行できます。
    1. **RHACM console → Infrastructure → Clusters → Cluster list**タブに移動します。
    2. 再配置操作を実行する前に、両方のマネージドクラスターのステータスを個別に確認してください。
- フェンシングを解除する前に、アプリケーションがクラスターからクリーンアップされたことを確認してください。

手順

1. ハブクラスターでフェンシングを無効にします。
    a. このクラスターの **DRCluster リソース** を編集し、<drcluster_name> を一意の名前に置き換えます。

```
$ oc edit drcluster <drcluster_name>
```

```
apiVersion: ramendr.openshift.io/v1alpha1
kind: DRCluster
metadata:
[...]
spec:
  cidrs:
  [...]
  ## Modify this line
```

```
  clusterFence: Unfenced
  [...]
[...]
```

出力例:

```
drcluster.ramendr.openshift.io/ocp4perf1 edited
```

b. **Fenced** であった OpenShift Container Platform ノードを正常に再起動します。リカバリーオーケストレーションの障害がさらに発生しないように、フェンシング解除後に I/O 操作を再開するには、再起動が必要です。ノードの正常な再起動 の手順に従って、クラスターのすべてのノードを再起動します。

注記

ノードで再起動して uncordon 操作を実行する前に、すべてのノードが最初に接続解除され、ドレインされていることを確認してください。

c. すべての OpenShift ノードが再起動され、**Ready** ステータスになったら、プライマリーマネージドクラスター (または Unfenced されたクラスター) でこのコマンドを実行して、すべての Pod が正常な状態であることを確認します。

```
oc get pods -A | egrep -v 'Running|Completed'
```

出力例:

```
NAMESPACE                                  NAME
READY   STATUS     RESTARTS      AGE
```

次のステップに進む前に、このクエリーの出力は 0 Pod である必要があります。

重要

ストレージ通信が切断されたために Pod がまだ異常な状態にある場合は、続行する前にトラブルシューティングを行って解決してください。ストレージクラスターは OpenShift の外部にあるため、OpenShift アプリケーションを正常に動作させるには、サイトの停止後にストレージクラスターを適切に復元する必要もあります。

または、OpenShift Web コンソールのダッシュボードと 概要 タブを使用して、アプリケーションと外部 ODF ストレージクラスターの正常性を評価することもできます。OpenShiftDataFoundation ダッシュボードの詳細は、**Storage → Data Foundation**に移動すると表示されます。

d. **Unfenced** クラスターが正常な状態であることを確認します。<drcluster_name> は、一意の名前に置き換えて、プライマリー管理クラスターのハブクラスターのフェンシングステータスを検証します。

```
$ oc get drcluster.ramendr.openshift.io <drcluster_name> -o jsonpath='{.status.phase}
{"\n"}'
```

出力例:

```
Unfenced
```

e. OpenShift Container Platform クラスターノードに属する IP がブロックリストに含まれていないことを確認します。

```
$ ceph osd blocklist ls
```

フェンシング中に追加された IP が表示されていないことを確認します。

2. ハブクラスターで、**Applications** に移動します。
3. アプリケーション行の最後にある **Actions** メニューをクリックして、使用可能なアクションのリストを表示します。
4. **Relocate application** をクリックします。
5. **Relocate application** モーダルが表示されたら、障害時に関連付けられたアプリケーションを再配置する **ポリシー** と **ターゲットクラスター** を選択します。
6. **Initiate** をクリックします。busybox リソースがターゲットクラスターに作成されました。
7. モーダルウィンドウを閉じ、Applications ページの **Data policy** 列を使用してステータスを追跡します。
8. アプリケーションのアクティビティーステータスが **Relocated** と表示されていることを確認します。
   a. **Applications → Overview** タブに移動します。
   b. **Data policy** 列で、ポリシーを適用したアプリケーションの **policy** リンクをクリックします。
   c. **Data policy** ポップオーバーで、1 つ以上のポリシー名と、アプリケーションで使用されているポリシーに関連付けられた再配置ステータスが表示されていることを確認します。

3.16. METRO-DR を使用した代替クラスターへの復旧

プライマリークラスターに障害が発生した場合、修復するか、既存のクラスターの回復を待つか、クラスターが交換不可能な場合はクラスター全体を交換するかのオプションが表示されます。このソリューションは、障害が発生したプライマリークラスターを新しいクラスターに置き換えるときにガイドし、この新しいクラスターへのフォールバック (再配置) を有効にします。

こちらの手順では、アプリケーションがインストールされ保護された後に RHACM マネージドクラスターを置き換える必要があることを前提としています。このセクションでは、RHACM マネージドクラスターを **置換クラスター** とし、置換されないクラスターを **存続クラスター**、新しいクラスターを **リカバリークラスター** とします。

前提条件

- Metro-DR 環境が、Red Hat Advance Cluster Management (RHACM) を使用してインストールされたアプリケーションで設定されている。
- アプリケーションをクラスター障害から保護するデータポリシーが割り当てられている。

手順

1. **ハブクラスター** で以下の手順を実行します。

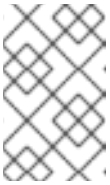
   a. CLI 端末を使用して **DRCluster** リソースを編集し、置換クラスターをフェンスします。**<drcluster_name>** は置換クラスター名に置き換えます。

   ```
   oc edit drcluster <drcluster_name>
   ```

   ```
   apiVersion: ramendr.openshift.io/v1alpha1
   kind: DRCluster
   metadata:
   [...]
   spec:
     ## Add or modify this line
     clusterFence: Fenced
     cidrs:
     [...]
   [...]
   ```

   b. RHACM コンソールを使用して、**アプリケーション** に移動し、障害が発生したクラスターから存続しているクラスターに保護されたすべてのアプリケーションをフェイルオーバーします。

   c. すべての保護されたアプリケーションが存続しているクラスター上で実行されていることを確認してください。

   注記

   各アプリケーションの **PROGRESSION** 状態は **Cleaning Up** として表示されます。代替クラスターがオフラインまたはダウンしている場合に、これは想定の動作です。。

2. 代替クラスターのフェンスを解除します。
   CLI 端末を使用して、DRCluster リソースを編集します。**<drcluster_name>** は置換クラスター名です。

   ```
   $ oc edit drcluster <drcluster_name>
   ```

   ```
   apiVersion: ramendr.openshift.io/v1alpha1
   kind: DRCluster
   metadata:
   [...]
   spec:
     ## Modify this line
     clusterFence: Unfenced
     cidrs:
     [...]
   [...]
   ```

3. 代替クラスターの DRCluster を削除します。

   ```
   $ oc delete drcluster <drcluster_name> --wait=false
   ```

注記

DRCluster は後のステップまで削除されないため、--wait=false を使用します。

4. 存続しているクラスター上の保護されたアプリケーションごとに、**ハブクラスター** の障害復旧を無効にします。

   a. アプリケーションごとに配置を編集し、存続したクラスターが選択されていることを確認します。

注記

サブスクリプションベースのアプリケーションの場合、関連する配置は、マネージドクラスターと同様に、ハブクラスター上の同じ namespace にあります。ApplicationSets ベースのアプリケーションの場合、関連する配置はハブクラスター上の **openshift-gitops** namespace にあります。

```
$ oc edit placement <placement_name> -n <namespace>
```

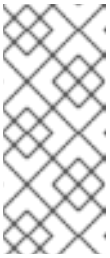
```
apiVersion: cluster.open-cluster-management.io/v1beta1
kind: Placement
metadata:
annotations:
  cluster.open-cluster-management.io/experimental-scheduling-disable: "true"
[...]
spec:
clusterSets:
- submariner
predicates:
- requiredClusterSelector:
    claimSelector: {}
    labelSelector:
      matchExpressions:
      - key: name
        operator: In
        values:
        - cluster1  <-- Modify to be surviving cluster name
[...]
```

   b. 保護されたアプリケーションの VolumeReplicationGroup ごとに、残りのクラスター上で次のコマンドを実行して、代替クラスターの **s3Profile** が削除されていることを確認します。

```
$ oc get vrg -n <application_namespace> -o jsonpath='{.items[0].spec.s3Profiles}' | jq
```

   c. 保護されたアプリケーションの **Placement** リソースが、保護されたアプリケーションから削除された存続クラスターと代替クラスター s3Profile を使用するようにすべて設定された後に、すべての **DRPlacementControl** リソースを **ハブクラスター** から削除する必要があります。

```
$ oc delete drpc <drpc_name> -n <namespace>
```



注記

サブスクリプションベースのアプリケーションの場合、関連する DRPlacementControl は、ハブクラスター上の管理対象クラスターと同じ namespace にあります。ApplicationSets ベースのアプリケーションの場合、関連付けられた DRPlacementControl はハブクラスターの **openshift-gitops** namespace にあります。

d. 次の手順に進む前に、すべての DRPlacementControl リソースが削除されていることを確認してください。このコマンドは、すべての namespace を対象にするクエリーです。リソースが見つからないはずです。

```
$ oc get drpc -A
```

e. 最後のステップでは、各アプリケーション **Placement** を編集し、**cluster.open-cluster-management.io/experimental-scheduling-disable: "true"** アノテーションを削除します。

```
$ oc edit placement <placement_name> -n <namespace>
```

```
apiVersion: cluster.open-cluster-management.io/v1beta1
kind: Placement
metadata:
annotations:
  ## Remove this annotation
  cluster.open-cluster-management.io/experimental-scheduling-disable: "true"
[...]
```

5. 存続したクラスター上で保護されたアプリケーションごとに、最後のステップとサブステップで説明したプロセスを繰り返します。保護されたアプリケーションの DR の無効化が完了しました。

6. ハブクラスターで、次のスクリプトを実行して、**存続するクラスター** と **ハブクラスター** からすべての障害復旧設定を削除します。

```
#!/bin/bash
secrets=$(oc get secrets -n openshift-operators | grep Opaque | cut -d" " -f1)
echo $secrets
for secret in $secrets
do
   oc patch -n openshift-operators secret/$secret -p '{"metadata":{"finalizers":null}}' --
type=merge
done
mirrorpeers=$(oc get mirrorpeer -o name)
echo $mirrorpeers
for mp in $mirrorpeers
do
   oc patch $mp -p '{"metadata":{"finalizers":null}}' --type=merge
   oc delete $mp
done
drpolicies=$(oc get drpolicy -o name)
echo $drpolicies
for drp in $drpolicies
do
```

```
    oc patch $drp -p '{"metadata":{"finalizers":null}}' --type=merge
    oc delete $drp
done
drclusters=$(oc get drcluster -o name)
echo $drclusters
for drp in $drclusters
do
    oc patch $drp -p '{"metadata":{"finalizers":null}}' --type=merge
    oc delete $drp
done
oc delete project openshift-operators
managedclusters=$(oc get managedclusters -o name | cut -d"/" -f2)
echo $managedclusters
for mc in $managedclusters
do
    secrets=$(oc get secrets -n $mc | grep multicluster.odf.openshift.io/secret-type | cut -d" " -
f1)
    echo $secrets
    for secret in $secrets
    do
        set -x
        oc patch -n $mc secret/$secret -p '{"metadata":{"finalizers":null}}' --type=merge
        oc delete -n $mc secret/$secret
    done
done

oc delete clusterrolebinding spoke-clusterrole-bindings
```

注記

このスクリプトでは、コマンド **oc delete project openshift-operators** を使用して、ハブクラスター上のこの namespace にある Disaster Recovery (DR) Operator を削除しました。この namespace に他の DR 以外の Operator がある場合は、OperatorHub から再度インストールする必要があります。

7. **openshift-operators** namespace が再び作成された後に、障害復旧メトリクスを収集するために monitoring ラベルを追加します。

```
$ oc label namespace openshift-operators openshift.io/cluster-monitoring='true'
```

8. 存続クラスターで、DR のインストール中に作成されたオブジェクトバケットが削除されていることを確認します。スクリプトによって削除されなかった場合は、オブジェクトバケットを削除します。DR に使用されるオブジェクトバケットの名前は、**odrbucket** で始まります。

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

9. RHACM コンソールで、**Infrastructure → Clusters ビュー** に移動します。

   a. 代替クラスターの割り当てを解除します。

   b. 新しい OpenShift クラスター (リカバリークラスター) を作成し、その新しいクラスターを RHACM コンソールにインポートします。手順は、クラスターの作成 および ハブクラスターへのターゲット管理対象クラスターのインポート を参照してください。

10. リカバリークラスターに OpenShift Data Foundation Operator をインストールし、それを存続クラスターと同じ外部 Ceph Storage システムに接続します。詳しい手順は、外部モードでの OpenShift Data Foundation のデプロイ を参照してください。

注記

OpenShift Data Foundation のバージョンが 4.15 (またはそれ以降) であり、同じバージョンの OpenShift Data Foundation が存続クラスター上にあることを確認してください。

11. ハブクラスターで、OperatorHub から ODF Multicluster Orchestrator Operator をインストールします。手順は、OpenShift Data Foundation Multicluster Orchestrator Operator のインストール の章を参照してください。

12. RHACM コンソールを使用して、**Data Services → Data policies** に移動します。

    a. **Create DRPolicy** を選択し、ポリシーに名前を付けます。

    b. **recovery cluster** と **surviving cluster** を選択します。

    c. ポリシーを作成します。手順は、ハブクラスターでの障害復旧ポリシーの作成 の章を参照してください。

    DRPolicy のステータスが **Validated** に変更された後にのみ、次の手順に進みます。

13. 代替クラスターに障害が発生する前に元々保護されていた、存続クラスター上のアプリケーションに DRPolicy を適用します。

14. 存続クラスター上で新しく保護されたアプリケーションを新しい回復 (プライマリー) クラスターに再配置します。RHACM コンソールを使用して、**Applications** メニューに移動して再配置を実行します。

3.17. RED HAT ADVANCED CLUSTER MANAGEMENT を使用したハブのリカバリー [テクノロジープレビュー]

セットアップにアクティブおよびパッシブの Red Hat Advanced Cluster Management for Kubernetes (RHACM) ハブクラスターがあり、アクティブハブがダウンしている場合は、パッシブハブを使用して、障害復旧が保護されたワークロードをフェイルオーバーまたは再配置できます。

重要

ハブリカバリーはテクノロジープレビュー機能で、テクノロジープレビューのサポート制限の対象となります。テクノロジープレビュー機能は、Red Hat 製品サポートのサービスレベルアグリーメント (SLA) の対象外であり、機能的に完全ではない場合があります。Red Hat は、実稼働環境でこれらを使用することを推奨していません。これらの機能は、近々発表予定の製品機能をリリースに先駆けてご提供することにより、お客様は機能性をテストし、開発プロセス中にフィードバックをお寄せいただくことができます。

詳細は、テクノロジープレビュー機能のサポート範囲 を参照してください。

3.17.1. パッシブハブクラスターの設定

アクティブハブがダウンしているかアクセスできない場合にハブリカバリーを実行するには、このセクションの手順に従ってパッシブハブクラスターを設定し、障害復旧保護されたワークロードをフェイルオーバーまたは再配置します。

手順

1. RHACM Operator と **MultiClusterHub** がパッシブハブクラスターにインストールされていることを確認します。手順については、RHACM インストールガイド を参照してください。Operator が正常にインストールされると、Web console update is available というメッセージを含むポップアップがユーザーインターフェイスに表示されます。このポップアップから **Refresh web console** をクリックして、反映するコンソールを変更します。

2. ハブをリカバリーする前に、バックアップとリストアを設定します。**RHACM ビジネス継続性** ガイドの バックアップと復元 を参照してください。

3. 復元の前に、マルチクラスターオーケストレーター (MCO) Operator を Red Hat OpenShift GitOps オペレーターとともにパッシブ RHACM ハブにインストールします。RHACM ハブをリストアする手順は、OpenShift Data Foundation Multicluster Orchestrator Operator のインストール を参照してください。

4. **Restore.cluster.open-cluster-management.io** リソースの **.spec.cleanupBeforeRestore** が **None** に設定されていることを確認します。詳細は、RHACM ドキュメントの バックアップの確認中にパッシブリソースを復元する を参照してください。

5. 以前のセットアップ中にクラスター間の SSL アクセスが手動で設定されていた場合は、クラスター間の SSL アクセスを再設定します。手順については、クラスター間での SSL アクセスの設定 の章を参照してください。

6. パッシブハブで、障害復旧メトリクスを収集するための監視ラベルを追加します。アラートの詳細は、障害復旧アラート を参照してください。

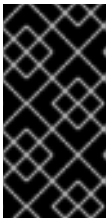
```
$ oc label namespace openshift-operators openshift.io/cluster-monitoring='true'
```

3.17.2. パッシブハブクラスターへの切り替え

アクティブハブがダウンしているか到達できない場合は、この手順を使用します。

手順

1. パッシブハブクラスターでバックアップを復元します。詳細は、バックアップからの ハブクラスターの復元 を参照してください。



重要

障害が発生したハブをパッシブインスタンスに復元すると、最後にスケジュールされたバックアップ時点のアプリケーションとその DR 保護状態のみが復元されます。最後のスケジュールされたバックアップの後に DR で保護されたアプリケーションは、新しいハブで再度保護する必要があります。

2. プライマリーおよびセカンダリー管理対象クラスターが RHACM コンソールに正常にインポートされ、アクセス可能であることを確認します。管理対象クラスターのいずれかがダウンしているか、アクセスできない場合は、正常にインポートされません。

3. DRPolicy 検証が成功するまで待ちます。

4. **DRPolicy** が正常に作成されたことを確認します。作成された各 DRPolicy リソースごとに **ハブクラスター** でこのコマンドを実行します。**<drpolicy_name>** は、一意の名前に置き換えてください。

```
$ oc get drpolicy <drpolicy_name> -o jsonpath='{.status.conditions[].reason}{"\n"}'
```

出力例:

```
Succeeded
```

5. アクティブハブクラスターで DR 監視ダッシュボードタブが有効になっている場合は、RHACM コンソールを更新して、そのタブにアクセスできるようにします。

6. アクティブハブクラスターのみがダウンしている場合は、ハブのリカバリーを実行し、パッシブハブでバックアップを復元して、ハブを復元します。管理対象クラスターにまだアクセスできる場合は、それ以上のアクションは必要ありません。

7. プライマリー管理対象クラスターがアクティブハブクラスターとともにダウンしている場合は、プライマリー管理対象クラスターからセカンダリー管理対象クラスターにワークロードをフェイルオーバーする必要があります。フェイルオーバーの手順については、ワークロードの種類に応じて、サブスクリプションベースのアプリケーション または ApplicationSet ベースのアプリケーション を参照してください。

8. フェイルオーバーが成功したことを確認します。プライマリー管理対象クラスターがダウンすると、ダウンした管理対象クラスターがオンラインに戻り、RHACM コンソールに正常にインポートされるまで、ワークロードの PROGRESSION ステータスは **Cleaning Up** フェーズになります。
パッシブハブクラスターで次のコマンドを実行して、PROGRESSION ステータスを確認します。

```
$ oc get drpc -o wide -A
```

出力例:

```
NAMESPACE            NAME                              AGE    PREFERREDCLUSTER
FAILOVERCLUSTER    DESIREDSTATE   CURRENTSTATE   PROGRESSION   START
TIME           DURATION       PEER READY
[...]
busybox              cephfs-busybox-placement-1-drpc       103m   cluster-1           cluster-2
Failover       FailedOver     Cleaning Up   2024-04-15T09:12:23Z                 False
busybox              cephfs-busybox-placement-1-drpc       102m   cluster-1
Deployed       Completed      2024-04-15T07:40:09Z   37.200569819s   True
[...]
```

第4章 OPENSHIFT DATA FOUNDATION の REGIONAL-DR ソリューション

4.1. REGIONAL-DR ソリューションのコンポーネント

Regional-DR は Red Hat Advanced Cluster Management for Kubernetes と OpenShift Data Foundation コンポーネントで設定され、Red Hat OpenShift Container Platform クラスター間でアプリケーションとデータの移動を提供します。

Red Hat Advanced Cluster Management for Kubernetes

Red Hat Advanced Cluster Management (RHACM) は、複数のクラスターとアプリケーションのライフサイクルを管理する機能を提供します。したがって、マルチクラスター環境でのコントロールプレーンとして機能します。

RHACM は 2 つの部分に分かれています。

- RHACM Hub: マルチクラスターコントロールプレーンで実行されるコンポーネント
- マネージドクラスター: マネージドクラスターで実行されるコンポーネント

この製品の詳細については、RHACM のドキュメント および RHACM のアプリケーションの管理 を参照してください。

OpenShift Data Foundation

OpenShift Data Foundation は、OpenShift Container Platform クラスターでステートフルなアプリケーション用のストレージをプロビジョニングし、管理する機能を提供します。

OpenShift Data Foundation はストレージプロバイダーとして Ceph をベースとしていて、そのライフサイクルは OpenShift Data Foundation コンポーネントスタックの Rook によって管理されます。Ceph-CSI は、ステートフルなアプリケーション用の永続ボリュームのプロビジョニングと管理を提供します。

OpenShift Data Foundation スタックは、以下の災害復旧機能で強化されました。

- OpenShift Data Foundation インスタンス (クラスター) 間でのミラーリングのために RBD ブロックプールを有効にします。
- RBD ブロックプール内の特定のイメージをミラーリングする機能
- Persistent Volume Claim (PVC) ミラーリングごとに管理する csi アドオンを提供します

OpenShift DR

OpenShift DR は、RHACM を使用して管理される一連のピア OpenShift クラスター全体でステートフルアプリケーションを設定および管理するための一連のオーケストレーターであり、永続ボリューム上のアプリケーションの状態のライフサイクルを調整するためのクラウドネイティブインターフェイスを提供します。これには以下が含まれます。

- OpenShift クラスター全体でアプリケーションとその状態関係を保護する
- アプリケーションとその状態をピアクラスターにフェイルオーバーする
- アプリケーションとその状態を以前にデプロイされたクラスターに再配置する

OpenShift DR は 3 つのコンポーネントに分類されます。

- **ODF マルチクラスターオーケストレーター**: マルチクラスターコントロールプレーン (RHACM ハブ) にインストールされ、メトロおよびリージョナル DR 関係のために OpenShift Data Foundation クラスターの設定とピアリングを調整します。

- **OpenShift DR Hub Operator**: ハブクラスターに ODF Multicluster Orchestrator インストールの一部として自動的にインストールされ、DR 対応アプリケーションのフェイルオーバーまたは再配置を調整します。

- **OpenShift DR Cluster Operator**: Metro および Regional DR 関係の一部である各マネージドクラスターに自動的にインストールされ、アプリケーションのすべての PVC のライフサイクルを管理します。

4.2. REGIONAL-DR デプロイメントワークフロー

このセクションでは、Red Hat OpenShift Data Foundation の最新バージョンを使用して、2 つの異なる OpenShift Container Platform クラスター間で Regional-DR 機能を設定およびデプロイするために必要な手順の概要を説明します。Red Hat Advanced Cluster Management (RHACM) をデプロイするには、2 つの管理対象クラスターに加えて、3 つ目の OpenShift Container Platform クラスターが必要になります。

インフラストラクチャーを設定するには、指定した順序で以下の手順を実行します。

1. DR ソリューションの一部であるハブ、プライマリー、およびセカンダリー Openshift Container Platform クラスターの 3 つの要件が満たされていることを確認します。Regional-DR を有効にするための要件を参照してください。

2. OpenShift Data Foundation operator をインストールし、プライマリーおよびセカンダリーのマネージドクラスターにストレージシステムを作成します。管理対象クラスターでの OpenShift Data Foundation クラスターの作成 を参照してください。

3. ハブクラスターに ODF マルチクラスターオーケストレーターをインストールします。ハブクラスターへの ODF Multicluster Orchestrator のインストール を参照してください。

4. ハブ、プライマリー、およびセカンダリークラスター間の SSL アクセスを設定します。クラスター間での SSL アクセスの設定 を参照してください。

5. プライマリークラスターとセカンダリークラスター全体で DR 保護を必要とするアプリケーションで使用する DRPolicy リソースを作成します。Creating Disaster Recovery Policy on Hub cluster を参照してください。

   注記

   複数のポリシーが存在する場合があります。

6. 以下を使用して災害復旧ソリューションをテストします。

   a. **サブスクリプションベース** のアプリケーション:

      - サブスクリプションベースのアプリケーションを作成します。サンプルアプリケーションの作成 を参照してください。

      - サンプルのサブスクリプションベースのアプリケーションを使用して、マネージドクラスター間でフェイルオーバーと再配置操作をテストします。サブスクリプションベースのアプリケーションのフェイルオーバー と サブスクリプションベースのアプリケーションの再配置 を参照してください。

b. **ApplicationSet ベース** のアプリケーション:

- サンプルアプリケーションを作成します。ApplicationSet ベースのアプリケーションの作成 を参照してください。
- マネージドクラスター間でサンプルアプリケーションを使用して、フェイルオーバーと再配置操作をテストします。ApplicationSet ベースのアプリケーションのフェイルオーバー と ApplicationSet ベースのアプリケーションの再配置 を参照してください。

4.3. REGIONAL-DR を有効にするための要件

Red Hat OpenShift Data Foundation でサポートされる障害復旧ソリューションをインストールするための前提条件は次のとおりです。

- 相互にネットワーク接続が可能な 3 つの OpenShift クラスターが必要です。
  - Red Hat Advanced Cluster Management (RHACM) for Kubernetes operator がインストールされている **ハブクラスター**。
  - OpenShift Data Foundation が実行されている **プライマリーマネージドクラスター**。
  - OpenShift Data Foundation が実行されている **セカンダリーマネージドクラスター**。

注記

ハブリカバリーセットアップを設定するには、パッシブハブとして機能する 4 番目のクラスターが必要です。プライマリー管理対象クラスター (Site-1) はアクティブ RHACM ハブクラスターと共存でき、パッシブハブクラスターはセカンダリー管理対象クラスター (Site-2) とともに配置できます。あるいは、アクティブな RHACM ハブクラスターを、サイト 1 のプライマリー管理対象クラスターまたはサイト 2 のセカンダリークラスターのいずれかの障害の影響を受けない中立サイト (サイト 3) に配置することもできます。この状況では、パッシブハブクラスターを使用する場合は、サイト 2 のセカンダリークラスターと一緒に配置できます。詳細は、ハブリカバリー用のパッシブハブクラスターの設定 を参照してください。

ハブリカバリーはテクノロジープレビュー機能で、テクノロジープレビューのサポート制限の対象となります。

- RHACM Operator と Multiclusterhub がハブクラスターにインストールされていることを確認します。手順については、RHACM インストールガイド を参照してください。
  Operator が正常にインストールされると、Web console update is available というメッセージを含むポップアップがユーザーインターフェイスに表示されます。このポップアップから **Refresh web console** をクリックして、反映するコンソールを変更します。

重要

アプリケーショントラフィックのルーティングとリダイレクトが適切に設定されていることを確認してください。

- ハブクラスターで以下を行います。
  - **All Clusters → Infrastructure → Clusters** に移動します。

- RHACM コンソールを使用して、**プライマリーマネージドクラスター** および **セカンダリーマネージドクラスター** をインポートまたは作成します。
- 環境に適したオプションを選択します。

手順は、クラスターの作成 および ハブクラスターへのターゲット管理対象クラスターのインポート を参照してください。

- RHACM Submariner アドオンを使用して、プライベート OpenShift クラスターおよびサービスネットワークを接続します。2 つのクラスターにサービスとクラスターのプライベートネットワークが重複していないことを確認します。それ以外の場合は、Submariner アドオンのインストール中に Globalnet が有効になっていることを確認してください。
マネージドクラスターごとに次のコマンドを実行して、Globalnet を有効にする必要があるかどうかを判断します。ここに示す例は、重複しないクラスターとサービスネットワークを対象としているため、Globalnet は有効になりません。

```
$ oc get networks.config.openshift.io cluster -o json | jq .spec
```

プライマリークラスターの出力例:

```
{
  "clusterNetwork": [
    {
      "cidr": "10.5.0.0/16",
      "hostPrefix": 23
    }
  ],
  "externalIP": {
    "policy": {}
  },
  "networkType": "OVNKubernetes",
  "serviceNetwork": [
    "10.15.0.0/16"
  ]
}
```

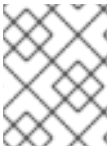
二次クラスターの出力例:

```
{
  "clusterNetwork": [
    {
      "cidr": "10.6.0.0/16",
      "hostPrefix": 23
    }
  ],
  "externalIP": {
    "policy": {}
  },
  "networkType": "OVNKubernetes",
  "serviceNetwork": [
    "10.16.0.0/16"
  ]
}
```

詳細は、Submariner ドキュメント を参照してください。

4.4. 管理対象クラスターでの OPENSHIFT DATA FOUNDATION クラスターの作成

2 つの OpenShift Container Platform クラスター間のストレージレプリケーションを設定するには、OpenShift Data Foundation Operator をインストールした後に OpenShift Data Foundation ストレージシステムを作成します。

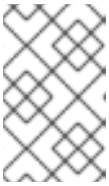


注記

インフラストラクチャー (AWS、VMware、BM、Azure など) に固有の OpenShift Data Foundation デプロイメントガイドと手順を参照してください。

手順

1. 各管理対象クラスターに最新の **OpenShift Data Foundation** クラスターをインストールして設定します。
   OpenShift Data Foundation のデプロイメントについては、インフラストラクチャー固有のデプロイメントガイド (AWS、VMware、ベアメタル、Azure など) を参照してください。

   注記

   ストレージクラスターの作成中に、**Data Protection** ステップで、**Prepare cluster for disaster recovery (Regional-DR only)** チェックボックスを選択する必要があります。

2. 以下のコマンドを使用して、各管理対象クラスターで OpenShift Data Foundation が正常にデプロイされたことを検証します。

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

   Multicloud Gateway (MCG) の場合:

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

   ステータス結果が **Primary managed cluster** と **Secondary managed cluster** の両方のクエリーに対して **Ready** である場合は、次の手順に進みます。

3. OpenShift Web コンソールで、**Installed Operators → OpenShift Data Foundation → Storage System → ocs-storagecluster-storagesystem → Resources** に移動し、**StorageCluster** の **Status** が **Ready** で、横に緑色のチェックマークがあることを確認します。

4. (オプション) Submariner のインストール時に Globalnet が有効になっていた場合は、OpenShift Data Foundation のインストールが完了した後に **StorageCluster** を編集します。Globalnet ネットワークの場合、**StorageCluster** yaml を手動で編集して **clusterID** を追加し、**enabled** を **true** に設定します。**<clustername>** は、インポートされた RHACM または新しく作成されたマネージドクラスターの名前に置き換えます。プライマリーマネージドクラスターとセカンダリーマネージドクラスターの両方で **StorageCluster** を編集します。

警告

Submariner のインストール時に Globalnet を有効にしていない限り、**StorageCluster** でこの変更を行わないでください。

```
$ oc edit storagecluster -o yaml -n openshift-storage
```

```
spec:
  network:
    multiClusterService:
      clusterID: <clustername>
      enabled: true
```

5. 上記の変更を行った後、
   a. OSD Pod が再起動し、OSD サービスが作成されるまで待ちます。
   b. すべての MONS がフェイルオーバーするまで待ちます。
   c. MONS サービスと OSD サービスがエクスポートされていることを確認します。

```
$ oc get serviceexport -n openshift-storage
```

```
NAME              AGE
rook-ceph-mon-d   4d14h
rook-ceph-mon-e   4d14h
rook-ceph-mon-f   4d14h
rook-ceph-osd-0   4d14h
rook-ceph-osd-1   4d14h
rook-ceph-osd-2   4d14h
```

   d. クラスターが **Ready** 状態にあり、クラスターの正常性が **Health ok** を示す緑色のチェックマークが付いていることを確認します。手順 3 を使用して確認します。

4.5. OPENSHIFT DATA FOUNDATION MULTICLUSTER ORCHESTRATOR OPERATOR のインストール

OpenShift Data Foundation のマルチクラスターオーケストレーターは、ハブクラスターの OpenShift Container Platform の OperatorHub からインストールされるコントローラーです。

手順

1. **ハブクラスター** で **OperatorHub** に移動し、キーワードフィルターを使用して **ODF Multicluster Orchestrator** を検索します。
2. **ODF Multicluster Orchestrator** タイルをクリックします。
3. すべてのデフォルト設定をそのままにして、**Install** をクリックします。

Operator のリソースが **openshift-operators** プロジェクトにインストールされ、すべての namespace で利用できることを確認してください。

注記

ODF Multicluster Orchestrator レーターは、依存関係として RHACM ハブクラスターに **Openshift DR ハブ Operator** もインストールします。

4. Operator **Pod** が **Running** 状態であることを確認します。**OpenShift DR Hub Operator** も同時に **openshift-operators** namespace にインストールされます。

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

出力例:

```
NAME                                  READY   STATUS      RESTARTS    AGE
odf-multicluster-console-6845b795b9-blxrn   1/1     Running     0           4d20h
odfmo-controller-manager-f9d9dfb59-jbrsd    1/1     Running     0           4d20h
ramen-hub-operator-6fb887f885-fss4w         2/2     Running     0           4d20h
```

4.6. クラスター全体での SSL アクセスの設定

プライマリークラスターとセカンダリークラスター 間のネットワーク (SSL) アクセスを設定して、メタデータを代替クラスターのマルチクラウドゲートウェイ (MCG) **object bucket** に安全なトランスポートプロトコルを使用して格納し、**ハブクラスター** に格納してオブジェクトバケットへのアクセスを検証できるようにします。

注記

すべての OpenShift クラスターが環境の署名済みの有効な証明書セットを使用してデプロイされる場合は、このセクションを省略できます。

手順

1. プライマリーマネージドクラスター の Ingress 証明書をデプロイメントし、出力を **primary.crt** に保存します。

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

2. セカンダリーマネージドクラスターの Ingress 証明書を抽出し、出力を **secondary.crt** に保存します。

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

3. リモートクラスターの証明書バンドルを保持する新しい **ConfigMap** ファイルを、ファイル名 **cm-clusters-crt.yaml** で作成します。

注記

この例のように、クラスターごとに 3 つ以上の証明書が存在する可能性があります。また、以前作成した **primary.crt** ファイルおよび **secondary.crt** ファイルから、証明書の内容をコピーして貼り付けた後に、証明書の内容が正しくインデントされていることを確認します。

```
apiVersion: v1
data:
  ca-bundle.crt: |
    -----BEGIN CERTIFICATE-----
    <copy contents of cert1 from primary.crt here>
    -----END CERTIFICATE-----

    -----BEGIN CERTIFICATE-----
    <copy contents of cert2 from primary.crt here>
    -----END CERTIFICATE-----

    -----BEGIN CERTIFICATE-----
    <copy contents of cert3 primary.crt here>
    -----END CERTIFICATE----

    -----BEGIN CERTIFICATE-----
    <copy contents of cert1 from secondary.crt here>
    -----END CERTIFICATE-----

    -----BEGIN CERTIFICATE-----
    <copy contents of cert2 from secondary.crt here>
    -----END CERTIFICATE-----

    -----BEGIN CERTIFICATE-----
    <copy contents of cert3 from secondary.crt here>
    -----END CERTIFICATE-----
kind: ConfigMap
metadata:
  name: user-ca-bundle
  namespace: openshift-config
```

4. **プライマリーマネージドクラスター**、**セカンダリーマネージドクラスター**、**ハブクラスター** で **ConfigMap** を作成します。

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

出力例:

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

5. **プライマリーマネージドクラスター**、**セカンダリーマネージドクラスター**、および **ハブクラスター** 上のデフォルトのプロキシーリソースにパッチを適用します。

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

出力例:

```
proxy.config.openshift.io/cluster patched
```

4.7. ハブクラスターでの障害復旧ポリシーの作成

Openshift 災害復旧ポリシー (DRPolicy) リソースは、災害復旧ソリューションに参加する OpenShift Container Platform クラスターと目的のレプリケーション間隔を指定します。DRPolicy は、ユーザーが障害復旧ソリューションを必要とするアプリケーションに適用できるクラスタースコープのリソースです。

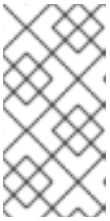
ODF MultiCluster Orchestrator Operator は、**Multicluster Web コンソール** を介して、各 **DRPolicy** および対応する **DRClusters** の作成を容易にします。

前提条件

- 2 つのマネージドクラスターの最小セットがあることを確認します。

手順

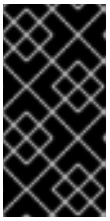
1. **OpenShift コンソール** で、**All Clusters → Data Services → Data policies** に移動します。
2. **DR ポリシーの作成** をクリックします。
3. **ポリシー名** を入力します。各 DRPolicy に一意の名前があることを確認します (例: **ocp4bos1-ocp4bos2-5m**)。
4. この新しいポリシーが関連付けられる管理対象クラスターのリストから 2 つのクラスターを選択します。



注記

クラスターを選択した後に "OSDs not migrated" というエラーメッセージが表示された場合は、次のステップに進む前に Migration of existing OSD to the optimized OSD in OpenShift Data Foundation for Regional-DR cluster のナレッジベース記事の手順に従ってください。

5. **レプリケーションポリシー** は、選択した OpenShift クラスターに基づいて自動的に **非同期** (async) に設定され、**同期スケジュール** オプションが使用可能になります。
6. **同期スケジュール** を設定します。



重要

必要なレプリケーション間隔ごとに、新しい **DRPolicy** を一意の名前 (例: **ocp4bos1-ocp4bos2-10m**) で作成する必要があります。同じクラスターを選択できますが、**同期スケジュール** は、分/時間/日単位の異なるレプリケーション間隔で設定できます。最小値は 1 分です。

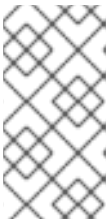
7. **Create** をクリックします。
8. **DRPolicy** が正常に作成されたことを確認します。作成された各 DRPolicy リソースごとに **ハブクラスター** でこのコマンドを実行します。**<drpolicy_name>** は、一意の名前に置き換えてください。

```
$ oc get drpolicy <drpolicy_name> -o jsonpath='{.status.conditions[].reason}{"\n"}'
```

出力例:

```
Succeeded
```

DRPolicy が作成されると、それに伴って 2 つの DRCluster リソースも作成されます。3 つのリソースすべてが検証され、ステータスが **Succeeded** と表示されるまで、最大 10 分かかる場合があります。



注記

DRPolicy で
は、**SchedulingInterval**、**ReplicationClassSelector**、**VolumeSnapshotClass Selector**、および **DRClusters** フィールドの値の編集はサポートされていません。

9. **ハブクラスター** から **プライマリーマネージドクラスター** と **セカンダリーマネージドクラスター** の両方へのオブジェクトバケットアクセスを確認します。

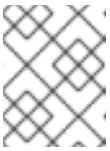
   a. ハブクラスター上の **DRClusters** の名前を取得します。

   ```
   $ oc get drclusters
   ```

   出力例:

   ```
   NAME      AGE
   ocp4bos1  4m42s
   ocp4bos2  4m42s
   ```

   b. 各マネージドクラスター上に作成された各バケットへの S3 アクセスを確認します。DRCluster 検証コマンドを使用します。**<drcluster_name>** は一意の名前に置き換えてください。

   注記

   DRCluster では、**Region** および **S3ProfileName** フィールド値の編集はサポートされていません。

   ```
   $ oc get drcluster <drcluster_name> -o jsonpath='{.status.conditions[2].reason}{"\n"}'
   ```

   出力例:

   ```
   Succeeded
   ```

   注記

   Hub cluster の両方の **DRClusters** に対してコマンドを実行してください。

10. **OpenShift DR Cluster Operator** のインストールが **プライマリー管理対象クラスター** および **セカンダリー管理対象** クラスターで成功したことを確認します。

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

出力例:

```
NAME                                                              DISPLAY                         VERSION
REPLACES   PHASE
clusterserviceversion.operators.coreos.com/odr-cluster-operator.v4.15.0          Openshift DR
Cluster Operator   4.15.0                    Succeeded
clusterserviceversion.operators.coreos.com/volsync-product.v0.8.0                VolSync
0.8.0                     Succeeded

NAME                                       READY   STATUS    RESTARTS   AGE
pod/ramen-dr-cluster-operator-6467cf5d4c-cc8kz   2/2     Running   0          3d12h
```

各管理対象クラスターの **OperatorHub** に **OpenShift DR Cluster Operator** が正常にインストールされていることを確認することもできます。

注記

最初の実行時に、VolSync Operator が自動的にインストールされます。VolSync は、CephFs ベースの PVC を保護するために、2 つのクラスター間でボリュームのレプリケーションを設定するために使用されます。レプリケーション機能はデフォルトで有効になっています。

11. **1 次管理対象クラスター** および **2 次管理** 対象クラスター上の OpenShift Data Foundation ミラーリング **デーモン** の正常性の状況を確認します。

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

出力例:

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

注意

daemon_health と **health** が **Warning** から **OK** になるまで、最大 10 分かかる場合があります。最終的にステータスが **OK** にならない場合は、RHACM コンソールを使用して、管理対象クラスター間の Submariner 接続がまだ正常な状態であることを確認します。すべての値が **OK** になるまで先に進まないでください。

4.8. 障害復旧ソリューションをテストするためのサンプルアプリケーションを作成する

OpenShift Data Foundation 障害復旧 (DR) ソリューションは、RHACM によって管理されるサブスクリプションベースおよびアプリケーションセットベースのアプリケーションの障害復旧をサポートします。詳細は、サブスクリプション と ApplicationSet のドキュメントを参照してください。

次のセクションでは、アプリケーションを作成して DRPolicy をアプリケーションに適用する方法について詳しく説明します。

- サブスクリプションベースのアプリケーション

cluster-admin パーミッションを持たない OpenShift ユーザーは、ナレッジ記事 で、障害復旧アクションを実行するために必要なパーミッションをアプリケーションユーザーに割り当てる方法を参照してください。

- ApplicationSet ベースのアプリケーション
  クラスター管理者権限を持たない OpenShift ユーザーは、ApplicationSet ベースのアプリケーションを作成できません。

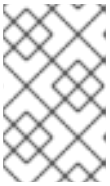
4.8.1. サブスクリプションベースのアプリケーション

4.8.1.1. サンプルのサブスクリプションベースのアプリケーションを作成する

プライマリーマネージドクラスター から **セカンダリーマネージドクラスター** への **failover** および **relocate** をテストするには、サンプルアプリケーションが必要です。

前提条件

- 一般消費用のアプリケーションを作成する場合は、アプリケーションが 1 つのクラスターのみにデプロイされるようにします。
- **busybox** というサンプルアプリケーションを例として使用します。
- アプリケーションのすべての外部ルートが、アプリケーションがフェイルオーバーまたは再配置されたときのトラフィックリダイレクト用に Global Traffic Manager (GTM) または Global Server Load Balancing (GLSB) サービスを使用して設定されていることを確認します。
- ベストプラクティスとして、一緒に属する Red Hat Advanced Cluster Management (RHACM) サブスクリプションをグループ化し、それらをグループとして DR で保護する単一の配置ルールを参照します。さらに、フェイルオーバーや再配置などの将来の DR アクションのために、その配置ルールを、サブスクリプションを論理的にグループ化する単一のアプリケーションとして作成します。



注記

関連のないサブスクリプションが配置アクションで同じ配置ルールを参照している場合、配置ルールを参照するすべてのサブスクリプションが DR ワークフローによって制御されるため、これらのサブスクリプションも DR で保護されます。

手順

1. ハブクラスターで、**Applications** に移動し、**Create application** をクリックします。
2. 種類は **Subscription**を選択します。
3. アプリケーションの **Name** (**busybox** など) および **Namespace** (**busybox-sample** など) を入力します。
4. Repository location for resources セクションで **Repository type Git** を選択します。
5. サンプルアプリケーションの github **Branch** および **Path** で、 Git リポジトリー URL を入力します。リソース **busybox** Pod および PVC が作成されます。
   - サンプルアプリケーションリポジトリーを **https://github.com/red-hat-storage/ocm-ramen-samples** として使用します。

- Branch を **release-4.15** として選択します。
- 次の **パス** のいずれかを選択します
  - RBD Regional-DR を使用するための **busybox-odr**。
  - CephFS Regional-DR を使用するための **busybox-odr-cephfs**。

6. **Deploy application resources on clusters with all specified labels**が表示されるまで、フォームを下にスクロールします。
   - グローバル **Cluster sets**、またはお使いの環境で、正しいマネージドクラスターが含まれるクラスターを選択します。
   - ラベル **<name>** を追加し、その値を **マネージドクラスター** 名に設定します。
7. 右上隅にある **作成** をクリックします。
   後続の画面で、**Topology** タブに移動します。アプリケーショントポロジーのチェックマークがすべて緑であることが確認できるはずです。

注記

詳細な情報を表示するには、トポロジー要素のいずれかをクリックすると、トポロジービューの右側にウィンドウが表示されます。

8. サンプルアプリケーションのデプロイを検証しています。
   busybox アプリケーションが優先クラスターにデプロイされたので、デプロイを検証できます。

   RHACM によって **busybox** がデプロイされたマネージドクラスターにログインします。

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

出力例:

```
NAME                      READY   STATUS    RESTARTS   AGE
pod/busybox-67bf494b9-zl5tr   1/1     Running   0          77s


NAME                                 STATUS   VOLUME                                     CAPACITY   ACCESS
MODES   STORAGECLASS                  AGE
persistentvolumeclaim/busybox-pvc   Bound    pvc-c732e5fe-daaf-4c4d-99dd-462e04c18412
5Gi        RWO            ocs-storagecluster-ceph-rbd   77s
```

4.8.1.2. サンプルアプリケーションにデータポリシーを適用する

前提条件

- Data ポリシーで参照されている両方の管理対象クラスターが到達可能であることを確認する。そうでない場合、両方のクラスターがオンラインになるまで、アプリケーションは災害復旧で保護されません。

手順

1. ハブクラスターで、**All Clusters** → **Applications** に移動します。
2. アプリケーションの最後にあるアクションメニューをクリックして、使用可能なアクションのリストを表示します。
3. **Manage data policy**→ **Assign data policy**をクリックします。
4. **Policy** を選択し、**Next** をクリックします。
5. **Application resource** を選択し、**PVC label selector** を使用して、選択したアプリケーションリソースの **PVC label** を選択します。

注記

選択したアプリケーションリソースに対して複数の PVC ラベルを選択できます。**Add application resource**オプションを使用して、複数のリソースを追加することもできます。

6. すべてのアプリケーションリソースを追加したら、**Next** をクリックします。
7. **Policy configuration details** を確認し、**Assign** をクリックします。新しく割り当てられたデータポリシーが **Manage data policy** モーダルリストビューに表示されます。
8. アプリケーションページで、割り当てられたポリシーの詳細を表示できることを確認します。
   a. アプリケーションページで、**Data policy** 列に移動し、**policy link** をクリックしてビューをデプロイメントします。
   b. 割り当てられたポリシーの数と、フェイルオーバーおよび再配置のステータスが表示されることを確認します。
   c. **View more details** をクリックして、アプリケーションで使用されているポリシーによる進行中のアクティビティーのステータスを表示します。
9. オプション: プライマリークラスター上の RADOS ブロックデバイス (RBD) **volumereplication** および **volumereplicationgroup** を確認します。

```
$ oc get volumereplications.replication.storage.openshift.io -A
```

出力例:

```
NAME           AGE     VOLUMEREPLICATIONCLASS                PVCNAME
DESIREDSTATE   CURRENTSTATE
busybox-pvc    2d16h   rbd-volumereplicationclass-1625360775   busybox-pvc      primary
Primary
```

```
$ oc get volumereplicationgroups.ramendr.openshift.io -A
```

出力例:

```
NAME           DESIREDSTATE   CURRENTSTATE
busybox-drpc   primary        Primary
```

10. オプション: CephFS volsync レプリケーションソースがプライマリークラスターで正常にセットアップされ、VolSync ReplicationDestination がフェイルオーバークラスターでセットアップされていることを確認します。

```
$ oc get replicationsource -n busybox-sample
```

出力例:

```
NAME           SOURCE         LAST SYNC              DURATION          NEXT SYNC
busybox-pvc    busybox-pvc    2022-12-20T08:46:07Z   1m7.794661104s    2022-12-
20T08:50:00Z
```

```
$ oc get replicationdestination -n busybox-sample
```

出力例:

```
NAME           LAST SYNC              DURATION          NEXT SYNC
busybox-pvc    2022-12-20T08:46:32Z   4m39.52261108s
```

4.8.2. ApplicationSet ベースのアプリケーション

4.8.2.1. ApplicationSet ベースのアプリケーションの作成

前提条件

- Red Hat OpenShift GitOps Operator が Hub クラスターにインストールされている。手順は、RHACM のドキュメント を参照してください。
- プライマリーとセカンダリーの両方のマネージドクラスターが GitOps に登録されていることを確認します。登録手順は、GitOps へのマネージドクラスターの登録 を参照してください。次に、両方のマネージドクラスターを登録するために **GitOpsCluster** リソースによって使用される配置に、クラスターが使用できない場合に対処するための許容機能があるかどうかを確認します。コマンド **oc get placement-name> -n openshift-gitops -o yaml** を使用して、次の許容範囲が Placement に追加されているかどうかを確認できます。

```
tolerations:
- key: cluster.open-cluster-management.io/unreachable
  operator: Exists
- key: cluster.open-cluster-management.io/unavailable
  operator: Exists
```

許容が追加されない場合は、 Red Hat Advanced Cluster Management および OpenShift GitOps のアプリケーション配置許容範囲の設定 を参照してください。

手順

1. ハブクラスターで、**All Clusters → Applications** に移動し、**Create application** をクリックします。
2. アプリケーションタイプを **Argo CD ApplicationSet - プッシュモデル** として選択します
3. 一般ステップ 1 で、**Application set name** を入力します。

4. **Argo サーバーopenshift-gitops** を選択し、**Requeue time** を **180** 秒に設定します。
5. **Next** をクリックします。
6. Repository location for resources セクションで **Repository type Git** を選択します。
7. サンプルアプリケーションの github Branch および Path で、 Git リポジトリー URL を入力します。リソース busybox Pod および PVC が作成されます。
    a. サンプルアプリケーションリポジトリーを https://github.com/red-hat-storage/ocm-ramen-samples として使用します。
    b. **Revision** を **release-4.15** として選択します。
    c. 次のパスのいずれかを選択します
        - RBD Regional-DR を使用するための **busybox-odr**。
        - CephFS Regional-DR を使用するための **busybox-odr-cephfs**。
8. **Remote namespace** の値 (例: busybox-sample) を入力し、 **Next** をクリックします。
9. **Sync policy** を選択し、**Next** をクリックします。
   1 つ以上のオプションを選択できます。
10. ラベル **<name>** を追加し、その値を **マネージドクラスター** 名に設定します。
11. **Next** をクリックします。
12. 設定の詳細を確認し、**Submit** をクリックします。

4.8.2.2. ApplicationSet ベースのサンプルアプリケーションにデータポリシーを適用する

前提条件

- Data ポリシーで参照されている両方の管理対象クラスターが到達可能であることを確認する。そうでない場合、両方のクラスターがオンラインになるまで、アプリケーションは災害復旧で保護されません。

手順

1. ハブクラスターで、**All Clusters → Applications** に移動します。
2. アプリケーションの最後にあるアクションメニューをクリックして、使用可能なアクションのリストを表示します。
3. **Manage data policy→ Assign data policy**をクリックします。
4. **Policy** を選択し、**Next** をクリックします。
5. **Application resource** を選択し、**PVC label selector** を使用して、選択したアプリケーションリソースの **PVC label** を選択します。

注記

選択したアプリケーションリソースに対して複数の PVC ラベルを選択できます。

6. すべてのアプリケーションリソースを追加したら、**Next** をクリックします。
7. **Policy configuration details** を確認し、**Assign** をクリックします。新しく割り当てられたデータポリシーが **Manage data policy** モーダルリストビューに表示されます。
8. アプリケーションページで、割り当てられたポリシーの詳細を表示できることを確認します。
   a. アプリケーションページで、**Data policy** 列に移動し、**policy link** をクリックしてビューをデプロイメントします。
   b. 割り当てられたポリシーの数と、フェイルオーバーおよび再配置のステータスが表示されることを確認します。
9. オプション: プライマリークラスター上の Rados ブロックデバイス (RBD) **volumereplication** および **volumereplicationgroup** を確認します。

```
$ oc get volumereplications.replication.storage.openshift.io -A
```

出力例:

```
NAME           AGE     VOLUMEREPLICATIONCLASS                PVCNAME
DESIREDSTATE   CURRENTSTATE
busybox-pvc    2d16h   rbd-volumereplicationclass-1625360775   busybox-pvc      primary
Primary
```

```
$ oc get volumereplicationgroups.ramendr.openshift.io -A
```

出力例:

```
NAME           DESIREDSTATE   CURRENTSTATE
busybox-drpc   primary        Primary
```

10. オプション: CephFS volsync レプリケーションソースがプライマリークラスターで正常にセットアップされ、VolSync ReplicationDestination がフェイルオーバークラスターでセットアップされていることを確認します。

```
$ oc get replicationsource -n busybox-sample
```

出力例:

```
NAME           SOURCE         LAST SYNC              DURATION          NEXT SYNC
busybox-pvc    busybox-pvc    2022-12-20T08:46:07Z   1m7.794661104s    2022-12-
20T08:50:00Z
```

```
$ oc get replicationdestination -n busybox-sample
```

出力例:

```
NAME          LAST SYNC              DURATION         NEXT SYNC
busybox-pvc   2022-12-20T08:46:32Z   4m39.52261108s
```

4.8.3. サンプルアプリケーションの削除

このセクションでは、RHACM コンソールを使用してサンプルアプリケーションの **Busybox** を削除する手順を説明します。

重要

DR で保護されたアプリケーションを削除する場合は、DRPolicy に属する両方のクラスターへのアクセスが必要です。これは、保護されたすべての API リソースと、それぞれの S3 ストア内のリソースが、DR 保護の削除の一環として確実に消去されるようにするためです。いずれかのクラスターへのアクセスが正常でない場合、ハブ上のアプリケーションの **DRPlacementControl** リソースを削除すると、Deleting の状態のままになります。

前提条件

- サンプルアプリケーションを削除する手順は、フェイルオーバーと再配置のテストが完了し、アプリケーションを RHACM とマネージドクラスターから削除する準備ができるまで実行しないでください。

手順

1. RHACM コンソールで、**Applications** に移動します。
2. 削除するサンプルアプリケーションを検索します (例: **busybox**)。
3. 削除するアプリケーションの横にある Action メニュー **(⋮)** をクリックします。
4. **Delete application** をクリックします。
   Delete application を選択すると、アプリケーション関連のリソースも削除すべきかどうかを求める新規画面が表示されます。
5. **Remove application related resources**チェックボックスを選択して、Subscription および PlacementRule を削除します。
6. **Delete** をクリックします。これにより、Primary マネージドクラスター (またはアプリケーションが実行しているクラスター) の busybox アプリケーションが削除されます。
7. RHACM コンソールを使用して削除されたリソースに加えて、**busybox** アプリケーションの削除後に **DRPlacementControl** が自動削除されない場合は、DRPlacementControl も削除します。
   a. ハブクラスターの OpenShift Web コンソールにログインし、プロジェクト **busybox-sample** の Installed Operators に移動します。
      ApplicationSet アプリケーションの場合は、プロジェクトを **openshift-gitops** として選択します。
   b. **OpenShift DR Hub Operator** をクリックした後、**DRPlacementControl** タブをクリックします。
   c. 削除する **busybox** アプリケーション DRPlacementControl の横にあるアクションメニュー **(⋮)** をクリックします。

d. **Delete DRPlacementControl** をクリックします。

e. **Delete** をクリックします。

注記

このプロセスを使用して、**DRPlacementControl** リソースでアプリケーションを削除できます。

4.9. マネージドクラスター間のサブスクリプションベースのアプリケーションフェイルオーバー

フェイルオーバーとは、プライマリークラスターに障害が発生した場合に、アプリケーションをプライマリークラスターからセカンダリークラスターに移行するプロセスです。フェイルオーバーにより、アプリケーションは最小限の中断でセカンダリークラスター上で実行できるようになりますが、情報に基づいてフェイルオーバーを決定すると、気づかない間に、プライマリークラスターからセカンダリークラスターへのレプリケーションの障害が発生した場合にデータが完全に失われるなど、悪影響が生じる可能性があります。最後にレプリケーションが成功してからかなりの時間が経過した場合は、障害が発生したプライマリーが回復するまで待つことを推奨します。

アプリケーションに関連付けられたすべての PVC に対して最後に成功したレプリケーションが発生してからの時間を反映するため、LastGroupSyncTime は重要なメトリック です。本質的には、プライマリークラスターとセカンダリークラスター間の同期の正常性を測定します。したがって、あるクラスターから別のクラスターへのフェイルオーバーを開始する前に、このメトリックを確認し、LastGroupSyncTime が、過去の妥当な時間内にある場合にのみフェイルオーバーを開始します。

注記

フェイルオーバー中に、フェイルオーバーされるクラスター上の Ceph RBD ミラーデプロイメントがスケールダウンされ、ストレージプロビジョニングとして Ceph RBD でバックされるボリュームのフェイルオーバーが正常に行われるようにします。

前提条件

- セットアップにアクティブおよびパッシブの RHACM ハブクラスターがある場合は、Red Hat Advanced Cluster Management を使用したハブのリカバリー を参照してください。
- プライマリークラスターが **Ready** 以外の状態にある場合は、更新に時間がかかる可能性があるため、クラスターの実際のステータスを確認してください。
  1. **RHACM console → Infrastructure → Clusters → Cluster list**タブに移動します。
  2. フェイルオーバー操作を実行する前に、両方のマネージドクラスターのステータスを個別に確認してください。
     ただし、フェイルオーバー先のクラスターが **Ready** 状態にある場合でも、フェイルオーバー操作は実行できます。
- ハブクラスターで次のコマンドを実行して、**lastGroupSyncTime** が現在時刻と比較して許容可能なデータ損失ウィンドウ内にあるかどうかを確認します。

```
$ oc get drpc -o yaml -A | grep lastGroupSyncTime
```

出力例:

```
[...]
lastGroupSyncTime: "2023-07-10T12:40:10Z"
```

手順

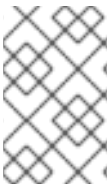
1. ハブクラスターで、**Applications** に移動します。
2. アプリケーション行の最後にある **Actions** メニューをクリックして、使用可能なアクションのリストを表示します。
3. **Failover application** をクリックします。
4. **Failover application** モーダルが表示されたら、障害時に関連付けられたアプリケーションがフェイルオーバーする **ポリシー** と **ターゲットクラスター** を選択します。
5. **Select subscription group** ドロップダウンをクリックして、デフォルトの選択を確認するか、この設定を変更します。
   デフォルトでは、アプリケーションリソースをレプリケートするサブスクリプショングループが選択されています。
6. **フェイルオーバーの準備** 状況を確認します。
   - ステータスが **Ready** で緑色のチェックマークが付いている場合は、ターゲットクラスターでフェイルオーバーを開始する準備ができていることを示しています。手順 7 に進みます。
   - ステータスが **Unknown** または **Not ready** の場合は、ステータスが **Ready** に変わるまで待ちます。
7. **Initiate** をクリックします。busybox アプリケーションが現在、**セカンダリーで管理されているクラスター** にフェイルオーバーしています。
8. モーダルウィンドウを閉じ、Applications ページの **Data policy** 列を使用してステータスを追跡します。
9. アプリケーションのアクティビティーステータスが **FailedOver** と表示されていることを確認します。
   a. **Applications → Overview** タブに移動します。
   b. **Data policy** 列で、ポリシーを適用したアプリケーションの **policy** リンクをクリックします。
   c. **Data policy** ポップオーバーで、**View more details** リンクをクリックします。
   d. アプリケーションで使用されているポリシーに関連付けられている 1 つ以上のポリシー名と進行中のアクティビティー (最終同期時間とアクティビティーステータス) が表示されることを確認します。

4.10. マネージドクラスター間の APPLICATIONSET ベースのアプリケーションフェイルオーバー

フェイルオーバーとは、プライマリークラスターに障害が発生した場合に、アプリケーションをプライマリークラスターからセカンダリークラスターに移行するプロセスです。フェイルオーバーにより、アプリケーションは最小限の中断でセカンダリークラスター上で実行できるようになりますが、情報に基づいてフェイルオーバーを決定すると、気づかない間に、プライマリークラスターからセカンダリーク

ラスターへのレプリケーションの障害が発生した場合にデータが完全に失われるなど、悪影響が生じる可能性があります。最後にレプリケーションが成功してからかなりの時間が経過した場合は、障害が発生したプライマリーが回復するまで待つことを推奨します。

アプリケーションに関連付けられたすべての PVC に対して最後に成功したレプリケーションが発生してからの時間を反映するため、LastGroupSyncTime は重要なメトリック です。本質的には、プライマリークラスターとセカンダリークラスター間の同期の正常性を測定します。したがって、あるクラスターから別のクラスターへのフェイルオーバーを開始する前に、このメトリックを確認し、LastGroupSyncTime が、過去の妥当な時間内にある場合にのみフェイルオーバーを開始します。



注記

フェイルオーバー中に、フェイルオーバーされるクラスター上の Ceph RBD ミラーデプロイメントがスケールダウンされ、ストレージプロビジョニングとして Ceph RBD でバックされるボリュームのフェイルオーバーが正常に行われるようにします。

前提条件

- セットアップにアクティブおよびパッシブの RHACM ハブクラスターがある場合は、Red Hat Advanced Cluster Management を使用したハブのリカバリー を参照してください。
- プライマリークラスターが **Ready** 以外の状態にある場合は、更新に時間がかかる可能性があるため、クラスターの実際のステータスを確認してください。
  1. **RHACM console → Infrastructure → Clusters → Cluster list**タブに移動します。
  2. フェイルオーバー操作を実行する前に、両方のマネージドクラスターのステータスを個別に確認してください。
     ただし、フェイルオーバー先のクラスターが **Ready** 状態にある場合でも、フェイルオーバー操作は実行できます。
- ハブクラスターで次のコマンドを実行して、**lastGroupSyncTime** が現在時刻と比較して許容可能なデータ損失ウィンドウ内にあるかどうかを確認します。

```
$ oc get drpc -o yaml -A | grep lastGroupSyncTime
```

出力例:

```
[...]
lastGroupSyncTime: "2023-07-10T12:40:10Z"
```

手順

1. ハブクラスターで、**Applications** に移動します。
2. アプリケーション行の最後にある **Actions** メニューをクリックして、使用可能なアクションのリストを表示します。
3. **Failover application** をクリックします。
4. **フェイルオーバーアプリケーション** モーダルが表示されたら、表示された詳細が正しいことを確認し、フェイルオーバーの **準備状況** のステータスを確認します。ステータスが **Ready** で緑色のチェックマークが付いている場合は、ターゲットクラスターでフェイルオーバーを開始する準備ができていることを示しています。

5. **Initiate** をクリックします。busybox リソースがターゲットクラスターに作成されました。

6. モーダルウィンドウを閉じ、Applications ページの **Data policy** 列を使用してステータスを追跡します。

7. アプリケーションのアクティビティーステータスが **FailedOver** と表示されていることを確認します。

   a. **Applications → Overview** タブに移動します。

   b. **Data policy** 列で、ポリシーを適用したアプリケーションの **policy** リンクをクリックします。

   c. **Data policy** ポップオーバーで、1 つ以上のポリシー名と、アプリケーションで使用されているポリシーに関連付けられた進行中のアクティビティーが表示されていることを確認します。

4.11. マネージドクラスター間でのサブスクリプションベースのアプリケーションの再配置

すべてのマネージドクラスターが使用可能になったら、アプリケーションを適切な場所に再配置します。

前提条件

- セットアップにアクティブおよびパッシブの RHACM ハブクラスターがある場合は、Red Hat Advanced Cluster Management を使用したハブのリカバリー を参照してください。

- プライマリークラスターが **Ready** 以外の状態にある場合は、更新に時間がかかる可能性があるため、クラスターの実際のステータスを確認してください。再配置は、プライマリークラスターと優先クラスターの両方が稼働している場合にのみ実行できます。

  1. **RHACM console → Infrastructure → Clusters → Cluster list**タブに移動します。

  2. 再配置操作を実行する前に、両方のマネージドクラスターのステータスを個別に確認してください。

- 現在の時刻と比較して、**lastGroupSyncTime** がレプリケーション間隔 (例: 5 分) 内にある場合に、再配置を実行します。これは、単一アプリケーションの目標復旧時間 (RTO) を最小限に抑えるために推奨されます。
  ハブクラスターで以下のコマンドを実行します。

  ```
  $ oc get drpc -o yaml -A | grep lastGroupSyncTime
  ```

  出力例:

  ```
  [...]
  lastGroupSyncTime: "2023-07-10T12:40:10Z"
  ```

  出力時刻 (UTC) と現在の時刻を比較して、すべての **lastGroupSyncTime** 値がアプリケーションレプリケーション間隔内にあることを検証します。そうでない場合は、すべての **lastGroupSyncTime** 値が true になるまで、再配置を待ちます。

手順

1. ハブクラスターで、**Applications** に移動します。
2. アプリケーション行の最後にある **Actions** メニューをクリックして、使用可能なアクションのリストを表示します。
3. **Relocate application** をクリックします。
4. **Relocate application** モーダルが表示されたら、障害時に関連付けられたアプリケーションを再配置する **ポリシー** と **ターゲットクラスター** を選択します。
5. デフォルトでは、アプリケーションリソースをデプロイするサブスクリプショングループが選択されています。**Select subscription group** ドロップダウンをクリックして、デフォルトの選択を確認するか、この設定を変更します。
6. **再配置の準備** 状況を確認します。
   - ステータスが **Ready** で緑色のチェックマークが付いている場合は、ターゲットクラスターで再配置を開始する準備ができていることを示しています。手順 7 に進みます。
   - ステータスが **Unknown** または **Not ready** の場合は、ステータスが **Ready** に変わるまで待ちます。
7. **Initiate** をクリックします。busybox リソースがターゲットクラスターに作成されました。
8. モーダルウィンドウを閉じ、Applications ページの **Data policy** 列を使用してステータスを追跡します。
9. アプリケーションのアクティビティーステータスが **Relocated** と表示されていることを確認します。
   a. **Applications → Overview** タブに移動します。
   b. **Data policy** 列で、ポリシーを適用したアプリケーションの **policy** リンクをクリックします。
   c. **Data policy** ポップオーバーで、**View more details** リンクをクリックします。
   d. アプリケーションで使用されているポリシーに関連付けられている 1 つ以上のポリシー名と進行中のアクティビティー (最終同期時間とアクティビティーステータス) が表示されることを確認します。

4.12. APPLICATIONSET ベースアプリケーションのマネージドクラスター間での再配置

すべてのマネージドクラスターが使用可能になったら、アプリケーションを適切な場所に再配置します。

前提条件

- セットアップにアクティブおよびパッシブの RHACM ハブクラスターがある場合は、Red Hat Advanced Cluster Management を使用したハブのリカバリー を参照してください。
- プライマリークラスターが **Ready** 以外の状態にある場合は、更新に時間がかかる可能性があるため、クラスターの実際のステータスを確認してください。再配置は、プライマリークラスターと優先クラスターの両方が稼働している場合にのみ実行できます。
  1. **RHACM console → Infrastructure → Clusters → Cluster list**タブに移動します。

2. 再配置操作を実行する前に、両方のマネージドクラスターのステータスを個別に確認してください。

- 現在の時刻と比較して、**lastGroupSyncTime** がレプリケーション間隔 (例: 5 分) 内にある場合に、再配置を実行します。これは、単一アプリケーションの目標復旧時間 (RTO) を最小限に抑えるために推奨されます。
  ハブクラスターで以下のコマンドを実行します。

```
$ oc get drpc -o yaml -A | grep lastGroupSyncTime
```

  出力例:

```
[...]
lastGroupSyncTime: "2023-07-10T12:40:10Z"
```

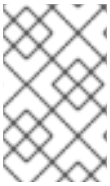
  出力時刻 (UTC) と現在の時刻を比較して、すべての **lastGroupSyncTime** 値がアプリケーションレプリケーション間隔内にあることを検証します。そうでない場合は、すべての **lastGroupSyncTime** 値が true になるまで、再配置を待ちます。

手順

1. ハブクラスターで、**Applications** に移動します。
2. アプリケーション行の最後にある **Actions** メニューをクリックして、使用可能なアクションのリストを表示します。
3. **Relocate application** をクリックします。
4. **Relocate application** モーダルが表示されたら、障害時に関連付けられたアプリケーションを再配置する **ポリシー** と **ターゲットクラスター** を選択します。
5. **Initiate** をクリックします。busybox リソースがターゲットクラスターに作成されました。
6. モーダルウィンドウを閉じ、Applications ページの **Data policy** 列を使用してステータスを追跡します。
7. アプリケーションのアクティビティーステータスが **Relocated** と表示されていることを確認します。
   a. **Applications → Overview** タブに移動します。
   b. **Data policy** 列で、ポリシーを適用したアプリケーションの **policy** リンクをクリックします。
   c. **Data policy** ポップオーバーで、1 つ以上のポリシー名と、アプリケーションで使用されているポリシーに関連付けられた再配置ステータスが表示されていることを確認します。

4.13. 障害復旧対応アプリケーションの RECOVERY POINT OBJECTIVE 値の表示

Recovery Point Objective (RPO) 値は、アプリケーションが現在アクティブなクラスターからそのピアへの永続データの最新の同期時間です。この同期時間は、フェイルオーバー中に失われたデータの期間を決定するのに役立ちます。



注記

この RPO 値は、フェイルオーバー中の Regional-DR にのみ適用されます。再配置では、すべてのピアクラスターが利用可能であるため、操作中のデータ損失はありません。

ハブクラスター上のワークロードについて、保護されているすべてのボリュームの Recovery Point Objective (RPO) 値を表示できます。

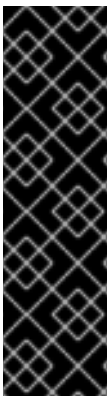
手順

1. ハブクラスターで、**Applications → Overview** タブに移動します。
2. **Data policy** 列で、ポリシーを適用したアプリケーションの **policy** リンクをクリックします。
   Data Policies モーダルページが表示され、各アプリケーションに適用されている障害復旧ポリシーの数と、フェイルオーバーおよび再配置の状況が示されます。
3. **Data Policies** モーダルページで、**View more details** リンクをクリックします。
   詳細な **Data policies** モーダルページが表示され、アプリケーションに適用されるポリシーに関連付けられたポリシー名と進行中のアクティビティー (最終同期、アクティビティーステータス) が表示されます。

   モーダルページで報告される **Last sync time** は、アプリケーションに対して DR 保護されているすべてのボリュームの最新の同期時間を表します。

4.14. RED HAT ADVANCED CLUSTER MANAGEMENT を使用したハブのリカバリー [テクノロジープレビュー]

セットアップにアクティブおよびパッシブの Red Hat Advanced Cluster Management for Kubernetes (RHACM) ハブクラスターがあり、アクティブハブがダウンしている場合は、パッシブハブを使用して、障害復旧が保護されたワークロードをフェイルオーバーまたは再配置できます。



重要

ハブリカバリーはテクノロジープレビュー機能で、テクノロジープレビューのサポート制限の対象となります。テクノロジープレビュー機能は、Red Hat 製品サポートのサービスレベルアグリーメント (SLA) の対象外であり、機能的に完全ではない場合があります。Red Hat は、実稼働環境でこれらを使用することを推奨していません。これらの機能は、近々発表予定の製品機能をリリースに先駆けてご提供することにより、お客様は機能性をテストし、開発プロセス中にフィードバックをお寄せいただくことができます。

詳細は、テクノロジープレビュー機能のサポート範囲 を参照してください。

4.14.1. パッシブハブクラスターの設定

アクティブハブがダウンしているかアクセスできない場合にハブリカバリーを実行するには、このセクションの手順に従ってパッシブハブクラスターを設定し、障害復旧保護されたワークロードをフェイルオーバーまたは再配置します。

手順

1. RHACM Operator と **MultiClusterHub** がパッシブハブクラスターにインストールされていることを確認します。手順については、RHACM インストールガイド を参照してください。

Operator が正常にインストールされると、Web console update is available というメッセージを含むポップアップがユーザーインターフェイスに表示されます。このポップアップから **Refresh web console** をクリックして、反映するコンソールを変更します。

2. ハブをリカバリーする前に、バックアップとリストアを設定します。**RHACM ビジネス継続性** ガイドの バックアップと復元 を参照してください。

3. 復元の前に、マルチクラスターオーケストレーター (MCO) Operator を Red Hat OpenShift GitOps オペレーターとともにパッシブ RHACM ハブにインストールします。RHACM ハブを復元する手順は、OpenShift Data Foundation Multicluster Orchestrator Operator のインストール を参照してください。

4. **Restore.cluster.open-cluster-management.io** リソースの **.spec.cleanupBeforeRestore** が **None** に設定されていることを確認します。詳細は、RHACM ドキュメントの バックアップの確認中にパッシブリソースを復元する を参照してください。

5. 以前のセットアップ中にクラスター間の SSL アクセスが手動で設定されていた場合は、クラスター間の SSL アクセスを再設定します。手順については、クラスター間での SSL アクセスの設定 の章を参照してください。

6. パッシブハブで、このコマンドを使用して、**openshift-operators** namespace にラベルを追加し、**VolumeSyncronizationDelay** アラートの基本的な監視を有効にします。アラートの詳細は、障害復旧アラート を参照してください。

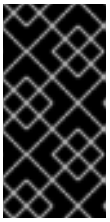
```
$ oc label namespace openshift-operators openshift.io/cluster-monitoring='true'
```

4.14.2. パッシブハブクラスターへの切り替え

アクティブハブがダウンしているか到達できない場合は、この手順を使用します。

手順

1. パッシブハブクラスターでバックアップを復元します。詳細は、バックアップからの ハブクラスターの復元 を参照してください。



重要

障害が発生したハブをパッシブインスタンスに復元すると、最後にスケジュールされたバックアップ時点のアプリケーションとその DR 保護状態のみが復元されます。最後のスケジュールされたバックアップの後に DR で保護されたアプリケーションは、新しいハブで再度保護する必要があります。

2. 管理対象クラスターがパッシブハブにインポートされると、Submariner が自動的にインストールされます。

3. プライマリーおよびセカンダリー管理対象クラスターが RHACM コンソールに正常にインポートされ、アクセス可能であることを確認します。管理対象クラスターのいずれかがダウンしているか、アクセスできない場合は、正常にインポートされません。

4. DRPolicy 検証が成功するまで待ちます。

5. **DRPolicy** が正常に作成されたことを確認します。作成された各 DRPolicy リソースごとに **ハブクラスター** でこのコマンドを実行します。**<drpolicy_name>** は、一意の名前に置き換えてください。

```
$ oc get drpolicy <drpolicy_name> -o jsonpath='{.status.conditions[].reason}{"\n"}'
```

出力例:

```
Succeeded
```

6. アクティブハブクラスターで DR 監視ダッシュボードタブが有効になっている場合は、RHACM コンソールを更新して、そのタブにアクセスできるようにします。

7. アクティブハブクラスターのみがダウンしている場合は、ハブのリカバリーを実行し、パッシブハブでバックアップを復元して、ハブを復元します。管理対象クラスターにまだアクセスできる場合は、それ以上のアクションは必要ありません。

8. プライマリー管理対象クラスターがアクティブハブクラスターとともにダウンしている場合は、プライマリー管理対象クラスターからセカンダリー管理対象クラスターにワークロードをフェイルオーバーする必要があります。フェイルオーバーの手順は、ワークロードの種類に応じて、サブスクリプションベースのアプリケーション または ApplicationSet ベースのアプリケーション を参照してください。

9. フェイルオーバーが成功したことを確認します。プライマリー管理対象クラスターがダウンすると、ダウンした管理対象クラスターがオンラインに戻り、RHACM コンソールに正常にインポートされるまで、ワークロードの PROGRESSION ステータスは **Cleaning Up** フェーズになります。
パッシブハブクラスターで次のコマンドを実行して、PROGRESSION ステータスを確認します。

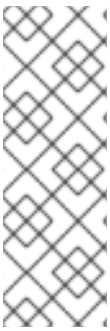
```
$ oc get drpc -o wide -A
```

出力例:

```
NAMESPACE            NAME                                AGE    PREFERREDCLUSTER
FAILOVERCLUSTER    DESIREDSTATE   CURRENTSTATE   PROGRESSION   START
TIME           DURATION       PEER READY
[...]
busybox              cephfs-busybox-placement-1-drpc       103m   cluster-1          cluster-2
Failover       FailedOver     Cleaning Up   2024-04-15T09:12:23Z                 False
busybox              cephfs-busybox-placement-1-drpc       102m   cluster-1
Deployed       Completed      2024-04-15T07:40:09Z   37.200569819s   True
[...]
```

第5章 OPENSHIFT DATA FOUNDATION のストレッチクラスターを使用した障害復旧

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デプロイメントできます。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。

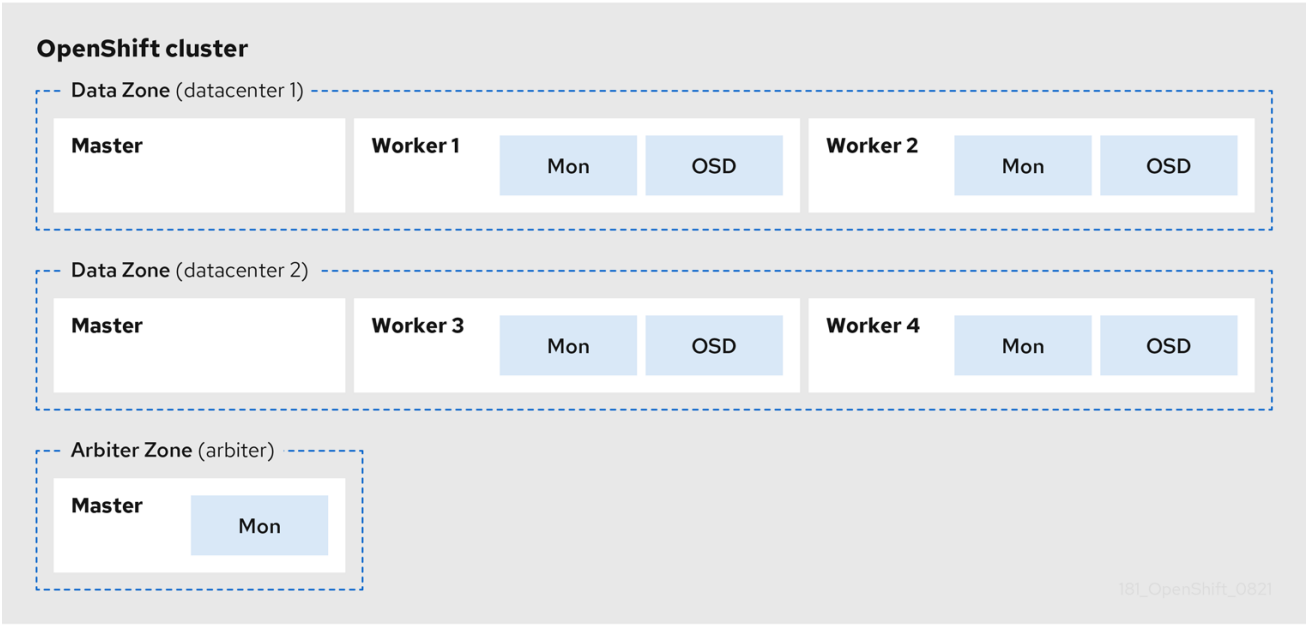


注記

ストレッチクラスターソリューションは、データボリュームを含むゾーン間の遅延が 10 ミリ秒の最大ラウンドトリップ時間 (RTT) を超えないデプロイメント向けに設計されています。Arbiter ノードは、etcd に指定されたレイテンシー要件に従います。詳細は、Guidance for Red Hat OpenShift Container Platform Clusters - Deployments Spanning Multiple Sites (Data Centers/Regions) を参照してください。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、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。

5.1. ストレッチクラスターを有効にするための要件

- 複数のサイトにまたがるデプロイメントの OpenShift Container Platform 要件に対応していることを確認してください。詳細は、knowledgebase article on cluster deployments spanning multiple sites を参照してください。
- 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。
- ストレッチクラスターソリューションは、ゾーン間の遅延が 10 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。

5.2. 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
```

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 Web コンソールからローカルストレージ Operator をインストール します。

5.3. 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. Channel を **stable** として更新します。
   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。

5.4. 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、**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. チャネルを **stable-4.15** に更新します。
   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** として選択します。
   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

- Operator が正常にインストールされると、**Web console update is available** メッセージを含むポップアップがユーザーインターフェイスに表示されます。このポップアップから **Refresh web console** をクリックして、反映するコンソールを変更します。
- Web コンソールに移動します。
  - Installed Operators に移動し、**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。
  - **Storage** に移動し、**Data Foundation** ダッシュボードが使用可能かどうかを確認します。

次のステップ

- OpenShift Data Foundation クラスターの作成

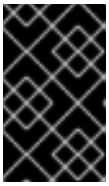
5.5. OPENSHIFT DATA FOUNDATION クラスターの作成

前提条件

- ストレッチクラスターを有効にするための要件セクション の要件をすべて満たしている。

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。



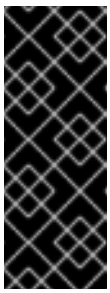
重要

Local Storage Operator がまだインストールされていない場合は、インストールするように求められます。**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。

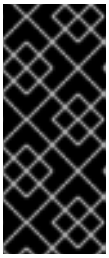
f. **Yes** をクリックして続行します。

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

a. **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 リストには、ストレージクラスに基づくノードが表示されます。

b. 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、ストレッチクラスターを有効にするための要件 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。

7. **Configure performance** でパフォーマンスプロファイルを選択します。
StorageSystems タブのオプションメニューから **Configure performance** オプションを使用して、デプロイ後にパフォーマンスプロファイルを設定することもできます。



重要

リソースプロファイルを選択する前に、クラスター内のリソースの現在の可用性を必ず確認してください。リソースが不十分なクラスターでより高いリソースプロファイルを選択すると、インストールが失敗する可能性があります。リソース要件の詳細は、パフォーマンスプロファイルのリソース要件 を参照してください。

a. **Next** をクリックします。

8. 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. 暗号化を有効にするには、**Enable data encryption for block and file storage**を選択します。

b. 以下の **Encryption level** (暗号化レベル) のいずれかを選択します。

- **Cluster-wide encryption**: クラスター全体 (ブロックおよびファイル) を暗号化します。
- **StorageClass encryption**: 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. オプション: **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** ドロップダウンリストから、**Vault** または **Thales CipherTrust Manager (using KMIP)** を選択します。**Vault** を選択した場合は、次の手順に進みます。**Thales CipherTrust Manager (using KMIP)** を選択した場合は、手順 iii に進みます。

ii. **認証方法**を選択します。

トークン認証方式の使用

- Vault ('https://<hostname or ip>') サーバーの一意の **Connection Name**、ホストの **Address**、**Port** 番号および **Token** を入力します。
- **Advanced Settings** をデプロイメントして、**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。
  - OpenShift Data Foundation 専用で固有のキーと値のシークレットパスを **Backend Path** に入力します。
  - オプション: **TLS Server Name** および **Vault Enterprise Namespace** を入力します。
  - PEM でエンコードされた、該当の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** を指定します。
  - **Save** をクリックして、手順 iv に進みます。

Kubernetes 認証方式の使用

- Vault ('https://<hostname or ip>') サーバーの一意の **Connection Name**、ホストの **Address**、**Port** 番号および **Role** 名を入力します。

- **Advanced Settings** をデプロイメントして、**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。
    - OpenShift Data Foundation 専用で固有のキーと値のシークレットパスを **Backend Path** に入力します。
    - 該当する場合は、**TLS Server Name** および **Authentication Path** を入力します。
    - PEM でエンコードされた、該当の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** を指定します。
    - **Save** をクリックして、手順 iv に進みます。

iii. **Thales CipherTrust Manager (using KMIP)** を KMS プロバイダーとして使用するには、次の手順に従います。

A. プロジェクト内のキー管理サービスの一意の **Connection Name** を入力します。

B. **Address** および **Port** セクションで、Thales CipherTrust Manager の IP と、KMIP インターフェイスが有効になっているポートを入力します。以下に例を示します。

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

C. **クライアント証明書**、**CA 証明書**、および **クライアント秘密鍵** をアップロードします。

D. StorageClass 暗号化が有効になっている場合は、上記で生成された暗号化および復号化に使用する一意の識別子を入力します。

E. **TLS Server** フィールドはオプションであり、KMIP エンドポイントの DNS エントリーがない場合に使用します。たとえば、**kmip_all_<port>.ciphertrustmanager.local** などです。

d. 単一のネットワークを使用している場合、**Network** はデフォルト (OVN) に設定されます。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は、カスタム (Multus) に切り替えて、次のいずれかを選択できます。

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

1 つの追加ネットワークインターフェイスのみを使用する場合は、パブリックネットワークインターフェイス (例: **ocs-public-cluster**) の単一の **NetworkAttachementDefinition** を選択し、Cluster Network Interface を空白のままにします。

a. **Next** をクリックします。

9. **Data Protection** ページで、**Next** をクリックします。

10. **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** をクリックして前の設定ページに戻ります。

11. **Create StorageSystem** 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** 完了で、横に緑色のチェックマークが付い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。

5.6. 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。

5.6.1. 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リスト表示します。

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5.1「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5.1 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) • **csi-addons-controller-manager- *** (任意のワーカーノードに 1 つの Pod)
Rook-ceph Operator	**rook-ceph-operator-*** (任意のワーカーノードに 1 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) • **noobaa-db-pg-*** (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)

コンポーネント	対応する 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)
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)

5.6.2. 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 → 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。

5.6.3. 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 → 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 card** 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。

重要

Multicloud Object Gateway には、データベースのコピー (NooBaa DB) が 1 つだけあります。つまり、NooBaa DB PVC が破損し、回復できない場合は、Multicloud Object Gateway に存在するアプリケーションデータが完全に失われる可能性があります。このため、Red Hat では NooBaa DB PVC のバックアップを定期的に取ることを推奨しています。NooBaa DB に障害が発生して回復できない場合は、最新のバックアップバージョンに戻すことができます。NooBaa DB をバックアップする手順は、こちらのナレッジベースの記事 の手順に従ってください。

5.6.4. 特定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**

5.7. ゾーン認識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、ストレッチクラスター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。パフォーマンスの低下の速度または量は、ゾーン間の待ち時間と、ストレージを使用するアプリケーションの動作 (大量の書き込みトラフィックなど) 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、必ずストレッチ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。

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。

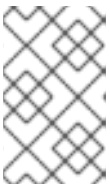
```
$ oc logs -f bc/file-uploader -n my-shared-storage
```

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。

5.7.1. インストール後のアプリケーションのスケーリング

手順

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。

5.7.2. 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
```

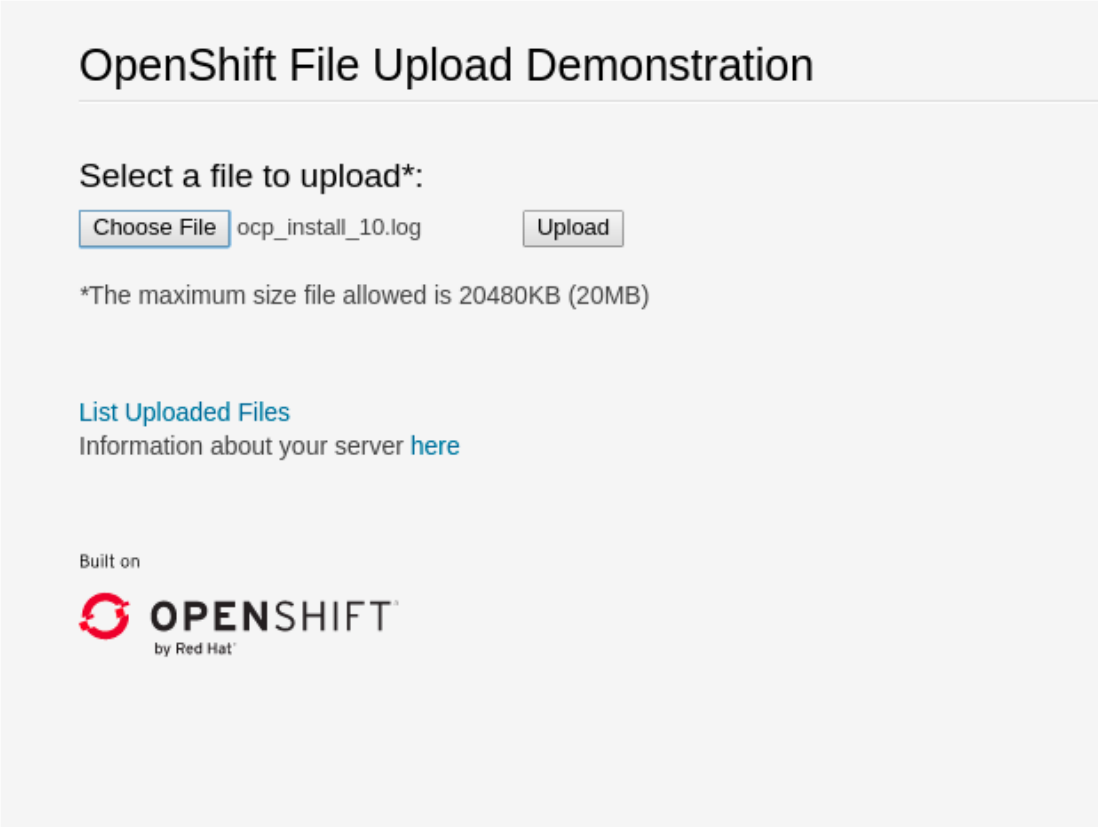
   b. 直前の手順のルートを使用して、ブラウザーを Web アプリケーションに指定します。Web アプリケーションはアップロードしたすべてのファイルをリスト表示し、新しいファイルをアップロードしたり、既存のデータをダウンロードする機能を提供します。今は何もありません。

   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. **Upload** 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リストを表示します。

注記

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

5.8. OPENSHIFT DATA FOUNDATION ストレッチクラスターのリカバリー

ストレッチクラスターの障害復旧のソリューションでは、完全または部分的なサイトの停止に直面しても回復性を提供することを考えると、アプリケーションとそのストレージのさまざまな復旧方法を理解することが重要です。

アプリケーションがどのように設計されているかによって、アクティブゾーンで再び利用できるようになるまでの時間が決まります。

サイトの停止に応じて、アプリケーションとそのストレージの復旧方法は異なります。復旧時間は、アプリケーションのアーキテクチャーによって異なります。復旧のさまざまな方法は以下のとおりです。

- RWX ストレージを使用したゾーン対応の HA アプリケーションの復旧
- RWX ストレージを使用した HA アプリケーションの復旧
- RWO ストレージを使用したアプリケーションの復旧
- StatefulSet Pod のリカバリー

5.8.1. ゾーンの障害について

このセクションでは、ゾーン障害は、ゾーン内のすべての OpenShift Container Platform、マスターノード、およびワーカーノードが 2 番目のデータゾーンのリソース (たとえば、電源がオフになっているノード) と通信しなくなった障害と見なされます。データゾーン間の通信がまだ部分的に機能している (断続的にアップまたはダウンしている) 場合、回復を成功させるには、クラスター、ストレージ、およびネットワーク管理者がデータゾーン間の通信パスを切断する必要があります。

重要

サンプルアプリケーションをインストールする場合は、OpenShift Container Platform ノード (OpenShift Data Foundation デバイスを使用するノード) の電源をオフにし、file-uploader アプリケーションが利用可能であることを検証するためにデータゾーンの失敗をテストし、新しいファイルをアップロードします。

5.8.2. RWX ストレージを使用したゾーン対応の HA アプリケーションの復旧

topologyKey: topology.kubernetes.io/zone を使用してデプロイされ、データゾーンごとに 1 つ以上のレプリカがスケジュールされており、共有ストレージ (ReadWriteMany (RWX) CephFS ボリューム) を使用しているアプリケーションは、障害のあったゾーンで数分後に自身で中断し、新しい Pod が起動し、ゾーンが復元されるまで Pending の状態で停止します。

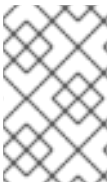
このタイプのアプリケーションの例は、ゾーン対応サンプルアプリケーションのインストール セクションを参照してください。

重要

ゾーンのリカバリー中に、CephFS ボリュームのマウント中にアプリケーション Pod がアクセス許可拒否エラーにより CrashLoopBackOff (CLBO) 状態になった場合は、Pod がスケジュールされているノードを再起動します。しばらく待ってから、Pod が再び実行されているかどうかを確認します。

5.8.3. RWX ストレージを使用した HA アプリケーションの復旧

topologyKey: kubernetes.io/hostname を使用している、またはトポロジー設定をまったく使用していないアプリケーションは、同じゾーンにあるすべてのアプリケーションレプリカに対する保護がありません。



注記

これは、**Pod** 仕様の **podAntiAffinity** および **topologyKey: kubernetes.io/hostname** でも発生する可能性があります。これは、この非アフィニティールールがホストベースであり、ゾーンベースではないためです。

これが発生し、すべてのレプリカが障害のあるゾーンにある場合、ReadWriteMany (RWX) ストレージを使用するアプリケーションはアクティブゾーンで回復するのに 6-8 分の時間がかかります。この一時停止は、障害が発生したゾーンの OpenShift Container Platform ノードが **NotReady**(60 秒) になり、デフォルトの Pod エビクションタイムアウトが期限切れ (300 秒) になるためです。

5.8.4. RWO ストレージを使用したアプリケーションの復旧

ReadWriteOnce (RWO) ストレージを使用するアプリケーションには、この Kubernetes の問題 で説明されている既知の動作があります。この問題のため、データゾーンに障害が発生した場合は、RWO ボリュームをマウントしているそのゾーンのアプリケーション Pod(例: **cephrbd** ベースのボリューム) は

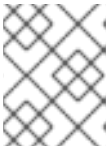
6〜8 分後に **Terminating** ステータスのままになり、手動の介入なしではアクティブゾーンに再作成されません。

ステータスが **NotReady** の OpenShift Container Platform ノードを確認します。ノードが OpenShift コントロールプレーンと通信できない問題が生じる可能性があります。ただし、ノードは永続ボリューム (PV) に対して I/O 操作を実行している可能性があります。

2 つの Pod が同じ RWO ボリュームに同時に書き込む場合は、データ破損のリスクが発生します。**NotReady** ノードのプロセスが終了するか、終了するまでブロックされていることを確認します。

ソリューションの例:

- 帯域外管理システムを使用してノードの電源をオフにし、確認を行うことは、プロセスを確実に終了させる例です。
- 障害が発生したサイトのノードによってストレージとの通信に使用されるネットワークルートを無効します。



注記

障害のあるゾーンまたはノードにサービスを復元する前に、PV が指定されたすべての Pod が正常に終了していることを確認します。

Terminating Pod がアクティブなゾーンで再作成されるようにするには、Pod を強制的に削除するか、関連付けられた PV でファイナライザーを削除します。これら 2 つのアクションのいずれかが完了すると、アプリケーション Pod がアクティブゾーンで再作成され、その RWO ストレージが正常にマウントされます。

Pod を強制的に削除

強制削除は、Pod が終了したという kubelet からの確認を待ちません。

```
$ oc delete pod <PODNAME> --grace-period=0 --force --namespace <NAMESPACE>
```

<PODNAME>

Pod の名前です。

<NAMESPACE>

プロジェクトの namespace です。

関連付けられた PV のファイナライザーの削除

Terminating Pod によってマウントされる Persistent Volume Claim (PVC) に関連付けられた PV を見つけ、**oc patch** コマンドを使用してファイナライザーを削除します。

```
$ oc patch -n openshift-storage pv/<PV_NAME> -p '{"metadata":{"finalizers":[]}}' --type=merge
```

<PV_NAME>

Pod の名前です。
関連付けられた PV を見つける簡単な方法として、Terminating Pod を記述することができます。複数割り当ての警告が表示される場合は、PV 名が警告に含まれるはずです (例: **pvc-0595a8d2-683f-443b-aee0-6e547f5f5a7c**)。

```
$ oc describe pod <PODNAME> --namespace <NAMESPACE>
```

<PODNAME>

Pod の名前です。

<NAMESPACE>

プロジェクトの namespace です。
出力例:

```
[...]
Events:
  Type     Reason                  Age   From                     Message
  ----     ------                  ----  ----                     -------
  Normal   Scheduled               4m5s  default-scheduler        Successfully assigned openshift-
storage/noobaa-db-pg-0 to perf1-mz8bt-worker-d2hdm
  Warning  FailedAttachVolume      4m5s  attachdetach-controller  Multi-Attach error for volume
"pvc-0595a8d2-683f-443b-aee0-6e547f5f5a7c" Volume is already exclusively attached to one
node and can't be attached to another
```

5.8.5. StatefulSet Pod のリカバリー

ステートフルセットの一部である Pod には、ReadWriteOnce (RWO) ボリュームをマウントする Pod と同様の問題があります。詳細は、Kubernetes リソース StatefulSet の考慮事項 で参照されます。

6-8 分後にアクティブなゾーンで再作成するために StatefulSet の Pod 部分を取得するには、RWO ボリュームを持つ Pod と同じ要件 (つまり、OpenShift Container Platform ノードの電源オフまたは通信が切断されている) で Pod を強制的に削除する必要があります。

第6章 障害復旧ヘルスの監視

6.1. 障害復旧のための監視の有効化

この手順を使用して、障害復旧セットアップの基本的な監視を有効にします。

手順

1. ハブクラスターでターミナルウィンドウを開きます。
2. 以下のラベルを **openshift-operator** namespace に追加します。

```
$ oc label namespace openshift-operators openshift.io/cluster-monitoring='true'
```

6.2. ハブクラスターでの障害復旧ダッシュボードの有効化

このセクションでは、ハブクラスター上で高度な監視を行うために障害復旧ダッシュボードを有効にする方法を説明します。

Regional-DR の場合、ダッシュボードには Operator の正常性、クラスターの正常性、メトリック、アラート、アプリケーション数の監視ステータスカードが表示されます。

Metro-DR の場合、Ramen セットアップの正常性とアプリケーション数のみを監視するようにダッシュボードを設定できます。

前提条件

- 以下がすでにインストールされている。
  - OpenShift Container Platform バージョン 4.15 がインストールされており、管理者権限がある。
  - コンソールプラグインが有効になっている ODF Multicluster Orchestrator。
  - Operator Hub の Red Hat Advanced Cluster Management for Kubernetes 2.10 (RHACM)。インストール方法は、RHACM のインストール を参照してください。
- RHACM で可観測性が有効になっている。可観測性の有効化に関するガイドライン を参照してください。

手順

1. ハブクラスターでターミナルウィンドウを開き、次の手順を実行します。
2. **observability-metrics-custom-allowlist.yaml** という名前の configmap ファイルを作成します。
   次の YAML を使用して、ハブクラスターの障害復旧メトリックをリスト表示できます。詳細は、カスタム指標の追加 を参照してください。ramen のメトリクスの詳細は、障害復旧のメトリクス を参照してください。

```
kind: ConfigMap
apiVersion: v1
metadata:
  name: observability-metrics-custom-allowlist
```

```
  namespace: open-cluster-management-observability
data:
  metrics_list.yaml: |
    names:
      - ceph_rbd_mirror_snapshot_sync_bytes
      - ceph_rbd_mirror_snapshot_snapshots
    matches:
      - __name__="csv_succeeded",exported_namespace="openshift-dr-system",name=~"odr-cluster-operator.*"
      - __name__="csv_succeeded",exported_namespace="openshift-operators",name=~"volsync.*"
```

3. **open-cluster-management-observability** namespace で次のコマンドを実行します。

```
$ oc apply -n open-cluster-management-observability -f observability-metrics-custom-allowlist.yaml
```

4. **observability-metrics-custom-allowlist yaml** が作成されると、RHACM はすべてのマネージドクラスターからリストされた OpenShift Data Foundation メトリックの収集を開始します。特定のマネージドクラスターを可観測性データの収集から除外するには、次のクラスターラベルを **clusters: observability: disabled** に追加します。

6.3. 災害復旧レプリケーション関係の正常性ステータスの表示

前提条件

災害復旧ダッシュボードの監視が有効になっていることを確認してください。手順は、ハブクラスターでの障害復旧ダッシュボードの有効化 の章を参照してください。

手順

1. ハブクラスターで、**すべてのクラスター** オプションが選択されていることを確認します。
2. コンソールを更新して、DR 監視ダッシュボードタブにアクセスできるようにします。
3. **Data Services** に移動し、**データポリシー** をクリックします。
4. 概要タブでは、operator、クラスター、アプリケーションの正常性ステータスを表示できます。緑色のチェックマークは、operator が実行中であり、使用可能であることを示します。
5. **障害復旧** タブをクリックして、DR ポリシーの詳細と接続されているアプリケーションのリストを表示します。

6.4. 障害復旧メトリック

これらは、prometheus によってスラッピングされる ramen メトリクスです。

- ramen_last_sync_timestamp_seconds
- ramen_policy_schedule_interval_seconds
- ramen_last_sync_duration_seconds
- ramen_last_sync_data_bytes

これらのメトリックは、Red Hat Advanced Cluster Management for Kubernetes (RHACM operator) がインストールされているハブクラスターから実行します。

最終同期のタイムスタンプ (秒単位)

これは、アプリケーションごとのすべての PVC の同期が最後に成功した時刻を示す秒単位の時間です。

メトリクス名

ramen_last_sync_timestamp_seconds

メトリクスのタイプ

ゲージ

Labels

- **ObjType**: オブジェクトのタイプ、ここでは DPPC
- **ObjName**: オブジェクトの名前。ここでは DRPC-Name です。
- **ObjNamespace**: drpc namespace
- **Policyname**: DRPolicy の名前
- **SchedulingInterval**: DRPolicy からのスケジュール間隔の値

メトリクス値

値は、DRPC ステータスの **lastGroupSyncTime** から取得される Unix 秒として設定されます。

ポリシースケジュール間隔 (秒)

これにより、DRPolicy からのスケジュール間隔が秒単位で得られます。

メトリクス名

ramen_policy_schedule_interval_seconds

メトリクスのタイプ

ゲージ

Labels

- **Policyname**: DRPolicy の名前

メトリクス値

これは、DRPolicy から取得したスケジュール間隔 (秒単位) に設定されます。

最後の同期期間 (秒)

これは、アプリケーションごとに全 PVC が正常に同期された最新の情報から同期する際にかかる最大時間を表します。

メトリクス名

ramen_last_sync_duration_seconds

メトリクスのタイプ

ゲージ

Labels

- **Obj_type**: オブジェクトのタイプ。ここでは DPPC です。
- **obj_name**: オブジェクトの名前。ここでは DRPC-Name です。
- **obj_namespace**: DRPC namespace。
- **scheduling_interval**: DRPolicy からのスケジュール間隔の値。

メトリクス値

値は、DRPC ステータスの **lastGroupSyncDuration** から取得されます。

最新の同期から転送された合計バイト数

これは、アプリケーションごとに全 PVC が正常に同期された最新の同期から転送された合計バイト数を表します。

メトリクス名

ramen_last_sync_data_bytes

メトリクスのタイプ

ゲージ

Labels

- **Obj_type**: オブジェクトのタイプ。ここでは DPPC です。
- **obj_name**: オブジェクトの名前。ここでは DRPC-Name です。
- **obj_namespace**: DRPC namespace。
- **scheduling_interval**: DRPolicy からのスケジュール間隔の値。

メトリクス値

値は、DRPC ステータスの **lastGroupSyncBytes** から取得されます。

6.5. 障害復旧アラート

このセクションでは、ディザスターリカバリー環境内の Red Hat OpenShift Data Foundation に関連してサポートされているすべてのアラートのリストを提供します。

記録ルール

- レコード: **ramen_sync_duration_seconds**

  式

  ```
  sum by (obj_name, obj_namespace, obj_type, job, policyname)(time() -
  (ramen_last_sync_timestamp_seconds > 0))
  ```

  目的

  ボリュームグループの最後の同期時刻と現在の時刻の間の時間間隔 (秒単位)。

- レコード: **ramen_rpo_difference**

  式

```
ramen_sync_duration_seconds{job="ramen-hub-operator-metrics-service"} /
on(policyname, job) group_left() (ramen_policy_schedule_interval_seconds{job="ramen-
hub-operator-metrics-service"})
```

目的

予想される同期遅延と、ボリュームレプリケーショングループによって発生する実際の同期遅延との差。

- レコード: **count_persistentvolumeclaim_total**

式

```
count(kube_persistentvolumeclaim_info)
```

目的

マネージドクラスターからのすべての PVC の合計。

アラート

- アラート: **VolumeSynchronizationDelay**

影響

Critical

目的

ボリュームレプリケーショングループによる実際の同期遅延は、予想される同期遅延の 3 倍になります。

YAML

```
alert: VolumeSynchronizationDela
expr: ramen_rpo_difference >= 3
for: 5s
labels:
  cluster: '{{ $labels.cluster }}'
  severity: critical
annotations:
  description: >-
    Syncing of volumes (DRPC: {{ $labels.obj_name }}, Namespace: {{
    $labels.obj_namespace }}) is taking more than thrice the scheduled
    snapshot interval. This may cause data loss and a backlog of replication
    requests.
  alert_type: DisasterRecovery
```

- アラート: **VolumeSynchronizationDelay**

影響

Warning

目的

ボリュームレプリケーショングループによる実際の同期遅延は、予想される同期遅延の 2 倍になります。

YAML

```
-
```

```
alert: VolumeSynchronizationDela
expr: ramen_rpo_difference > 2 and ramen_rpo_difference < 3
for: 5s
labels:
  cluster: '{{ $labels.cluster }}'
  severity: critical
annotations:
  description: >-
    Syncing of volumes (DRPC: {{ $labels.obj_name }}, Namespace: {{
    $labels.obj_namespace }}) is taking more than twice the scheduled
    snapshot interval. This may cause data loss and a backlog of replication
    requests.
  alert_type: DisasterRecovery
```

第7章 障害復旧のトラブルシューティング

7.1. METRO-DR のトラブルシューティング

7.1.1. フェイルオーバー後にステートフルセットアプリケーションがスタックする

問題

優先クラスターへの再配置中に、DRPlacementControl が停止し、PROGRESSION に "MovingToSecondary" と報告されます。
以前は、Kubernetes v1.23 より前のバージョンでは、Kubernetes コントロールプレーンが StatefulSet 用に作成された PVC をクリーンアップしませんでした。このアクティビティーは、クラスター管理者または StatefulSet を管理するソフトウェア operator に任されていました。これが原因で、Pod が削除されたときに、StatefulSet の PVC はそのまま残されました。これにより、Ramen がアプリケーションを優先クラスターに再配置できなくなります。

解決方法

1. ワークロードが StatefulSets を使用していて、再配置が停止し、PROGRESSION に "MovingToSecondary" と報告される場合は、次を実行します。

```
$ oc get pvc -n <namespace>
```

2. StatefulSet に属するnamespace の制限付き PVC ごとに、次を実行します。

```
$ oc delete pvc <pvcname> -n namespace
```

すべての PVC が削除されると、ボリュームレプリケーショングループ (VRG) はセカンダリーに移行してから削除されます。

3. 以下のコマンドを実行します。

```
$ oc get drpc -n <namespace> -o wide
```

数秒から数分後、PROGRESSION に "Completed" と報告され、再配置が完了します。

結果

ワークロードが優先クラスターに再配置されます。

BZ リファレンス: 2118270

7.1.2. DR ポリシーが同じ namespace 内のすべてのアプリケーションを保護する

問題

DR ポリシーで使用するアプリケーションが 1 つだけ選択されている場合でも、同じnamespace 内のすべてのアプリケーションが保護されます。これにより、複数のワークロードで **DRPlacementControl spec.pvcSelector** に一致する PVC が生成されるか、すべてのワークロードでセレクターが欠落している場合、レプリケーション管理が各 PVC を複数回管理し、個々の **DRPlacementControl** アクションに基づいてデータの破損または無効な操作を引き起こす可能性があります。

解決方法

ワークロードに属する PVC に一意のラベルを付け、選択したラベルを DRPlacementControl

spec.pvcSelector として使用して、どの DRPlacementControl がネームスペース内の PVC のどのサブセットを保護および管理するかを明確にします。ユーザーインターフェイスを使用して DRPlacementControl の **spec.pvcSelector** フィールドを指定することはできません。したがって、そのようなアプリケーションの DRPlacementControl を削除し、コマンドラインを使用して作成する必要があります。

BZ リファレンス: [2128860]

7.1.3. アプリケーションのフェイルバック中に Relocating 状態でスタックする

問題

この問題は、アプリケーションのフェイルオーバーおよびフェイルバックを実行した後 (すべてのノードまたはクラスターが稼働している) に発生する可能性があります。フェールバックを実行すると、アプリケーションが **Relocating** 状態でスタックし、PV リストアの完了に対して **Waiting** というメッセージが表示されます。

解決方法

S3 クライアントまたは同等のサービスを使用して、重複する PV オブジェクトを s3 ストアからクリーンアップします。タイムスタンプがフェイルオーバーまたは再配置の時刻に近いものだけを保持します。

BZ リファレンス: 2120201

7.1.4. 再配置またはフェイルバックが Initiating 状態でスタックする可能性がある

問題

プライマリークラスターがダウンして、セカンダリークラスターがダウンしている間にオンラインに戻ると、**relocate** または **failback** が **Initiating** の状態でスタックする可能性があります。

解決方法

この状況を回避するには、古いアクティブなハブからマネージドクラスターへの全アクセスを遮断します。
あるいは、ワークロードを移動する前またはクリーンアップフェーズにあるときに、古いアクティブハブクラスター上の ApplicationSet コントローラーをスケールダウンすることもできます。

以前のアクティブなハブで、以下のコマンドを使用して 2 つのデプロイメントをスケールダウンします。

```
$ oc scale deploy -n openshift-gitops-operator openshift-gitops-operator-controller-manager --replicas=0

$ oc scale statefulset -n openshift-gitops openshift-gitops-application-controller --replicas=0
```

BZ リファレンス: [2243804]

7.2. REGIONAL-DR のトラブルシューティング

7.2.1. rbd-mirror デーモンのヘルスが警告状態になる

問題

ミラーサービス **::get_mirror_service_status** が **Ceph** モニターを呼び出して **rbd-mirror** のサービスステータスを取得すると、WARNING が報告されるケースが多数あるようです。

ネットワークの切断後、**rbd-mirror** デーモンのヘルスは **warning** 状態になりますが、両方の管理対象クラスター間の接続は良好です。

解決方法

ツールボックスで次のコマンドを実行し、**leader:false** を探します。

```
rbd mirror pool status --verbose ocs-storagecluster-cephblockpool | grep 'leader:'
```

出力に次のように表示される場合:

leader: false	これは、デーモンの起動に問題があることを示しており、最も可能性の高い根本原因は、セカンダリークラスターへの確実な接続の問題である可能性があります。 回避策: Pod を削除するだけで **rbd-mirror** Pod を別のノードに移動し、別のノードで再スケジュールされていることを確認します。
leader: true または出力なし	Red Hat サポートに連絡してください。

BZ リファレンス: 2118627

7.2.2. 宛先ホスト名を解決できないため、volsync-rsync-src Pod がエラー状態になる

問題

VolSync ソース Pod は、VolSync 宛先 Pod のホスト名を解決できません。VolSync Pod のログには、次のログスニペットのようなエラーメッセージが長時間にわたって一貫して表示されます。

```
$ oc logs -n busybox-workloads-3-2 volsync-rsync-src-dd-io-pvc-1-p25rz
```

出力例

```
VolSync rsync container version: ACM-0.6.0-ce9a280
Syncing data to volsync-rsync-dst-dd-io-pvc-1.busybox-workloads-3-2.svc.clusterset.local:22 ...
ssh: Could not resolve hostname volsync-rsync-dst-dd-io-pvc-1.busybox-workloads-3-
2.svc.clusterset.local: Name or service not known
```

解決方法

両方のノードで **submariner-lighthouse-agent** を再起動します。

```
$ oc delete pod -l app=submariner-lighthouse-agent -n submariner-operator
```

7.2.3. フェイルオーバー後に古いプライマリーのマネージドクラスターが復元された後に、ApplicationSet ワークロードのクリーンアップとデータ同期が停止したままになる

問題

ハブクラスターに障害が発生した場合に、マネージドクラスターへの ApplicationSet ベースのワークロードのデプロイメントは、ガベージコレクションされません。ワークロードは残りのマネージ

ドクラスターにフェイルオーバーされている間、スタンバイハブクラスターに復元されます。ワークロードのフェイルオーバー元のクラスターは、新しく回復されたスタンバイハブに再び参加します。
したがって、Regional DRPolicy で DR 保護されている ApplicationSet は、VolumeSynchronizationDelay アラートの起動を開始します。さらに、このような DR で保護されたワークロードは、2 つのクラスター間でデータが同期されていないため、ピアクラスターにフェイルオーバーしたり、ピアクラスターに再配置したりできません。

解決方法

回避策として、新しく回復したハブからワークロードのフェイルオーバーが実行された後にハブに再参加したマネージドクラスター上で孤立したワークロードリソースを **openshift-gitops** Operator が所有できる必要があります。これを行うには、次の手順を実行します。

1. **openshift-gitops** namespace のハブクラスター上の ArgoCD ApplicationSet リソースによって使用されている配置を決定します。

2. このフィールドの ApplicationSet の配置ラベル値 (**spec.generators.clusterDecisionResource.labelSelector.matchLabels**) を検査します。これは、配置リソース **<placement-name>** の名前になります。

3. ApplicationSet で参照される **Placement** に **PlacemenDecision** が存在することを確認します。

   ```
   $ oc get placementdecision -n openshift-gitops --selector cluster.open-cluster-
   management.io/placement=<placement-name>
   ```

   これにより、現在必要なフェイルオーバークラスターにワークロードを配置する 1 つの **PlacementDecision** が生成されます。

4. クリーンアップする必要があるクラスターを指す ApplicationSet の新しい **PlacementDecision** を作成します。
   以下に例を示します。

   ```
   apiVersion: cluster.open-cluster-management.io/v1beta1
   kind: PlacementDecision
   metadata:
     labels:
       cluster.open-cluster-management.io/decision-group-index: "1" # Typically one higher
   than the same value in the esisting PlacementDecision determined at step (2)
       cluster.open-cluster-management.io/decision-group-name: ""
       cluster.open-cluster-management.io/placement: cephfs-appset-busybox10-placement
     name: <placemen-name>-decision-<n> # <n> should be one higher than the existing
   PlacementDecision as determined in step (2)
     namespace: openshift-gitops
   ```

5. 新たに作成した **PlacementDecision** を **subresource** のステータスに更新します。

   ```
   decision-status.yaml:
   status:
     decisions:
     - clusterName: <managedcluster-name-to-clean-up> # This would be the cluster from
   where the workload was failed over, NOT the current workload cluster
       reason: FailoverCleanup
   ```

```
$ oc patch placementdecision -n openshift-gitops <placemen-name>-decision-<n> --
patch-file=decision-status.yaml --subresource=status --type=merge
```

6. ApplicationSet のアプリケーションリソースが必要なクラスターに配置されていることを確認します。

```
$ oc get application -n openshift-gitops  <applicationset-name>-<managedcluster-name-
to-clean-up>
```

出力で、SYNC STATUS が **Synced** と表示され、HEALTH STATUS が **Healthy** として表示されているかどうかを確認します。

7. 手順 (3) で作成した PlacementDecision を削除します。これにより、ArgoCD は <managedcluster-name-to-clean-up> 上のワークロードリソースをガベージコレクションできるようになります。

```
$ oc delete placementdecision -n openshift-gitops <placemen-name>-decision-<n>
```

Regional DRPolicy で DR 保護されている ApplicationSet は、**VolumeSynchronizationDelay** アラートの実行を停止します。

BZ リファレンス: [2268594]

7.3. ARBITER を使用した 2 サイトストレッチクラスターのトラブルシューティング

7.3.1. ゾーン回復後の ContainerCreating 状態でスタックしたワークロード Pod の回復

問題

完全なゾーン障害とリカバリーを実行した後、ワークロード Pod が以下のいずれかのエラーで **ContainerCreating** 状態のままになることがあります。

- MountDevice failed to create newCsiDriverClient: driver name openshift-storage.rbd.csi.ceph.com not found in the list of registered CSI drivers
- MountDevice failed for volume **<volume_name>** : rpc error: code = Aborted desc = an operation with the given Volume ID **<volume_id>** already exists
- MountVolume.SetUp failed for volume **<volume_name>** : rpc error: code = Internal desc = staging path **<path>** for volume **<volume_id>** is not a mountpoint

解決方法

ワークロード Pod が上記のエラーのいずれかでスタックしている場合は、以下の回避策を実行してください。

- **ceph-fs** ワークロードが **ContainerCreating** でスタックしている場合:
  1. スタックした Pod がスケジュールされているノードを再起動します。
  2. これらのスタックした Pod を削除します。
  3. 新規 Pod が実行されていることを確認します。

- ceph-rbd ワークロードが **ContainerCreating** でスタックし、しばらくしても自己回復しない場合:

  1. スタックした Pod がスケジュールされているノードで csi-rbd プラグイン Pod を再起動します。

  2. 新規 Pod が実行されていることを確認します。