Red Hat OpenShift Data Foundation 4.9

Red Hat OpenShift Data Foundation アーキテクチャー

OpenShift Data Foundation アーキテクチャーと、コンポーネントおよびサービスが実行するロールの概要

Last Updated: 2022-07-10

Red Hat OpenShift Data Foundation 4.9 Red Hat OpenShift Data Foundation アーキテクチャー

OpenShift Data Foundation アーキテクチャーと、コンポーネントおよびサービスが実行するロールの概要

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

法律上の通知

概要

本書では、OpenShift Data Foundation アーキテクチャーの概要を説明します。

目次

前書き

本書では、OpenShift Data Foundation アーキテクチャーの概要を説明します。

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、弊社 の CTO、Chris Wright のメッセージ を参照してください。

RED HAT ドキュメントへのフィードバック (英語のみ)

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

- 特定の部分についての簡単なコメントをお寄せいただく場合は、以下をご確認ください。
  1. ドキュメントの表示が **Multi-page HTML** 形式になっていて、ドキュメントの右上端に **Feedback** ボタンがあることを確認してください。
  2. マウスカーソルで、コメントを追加する部分を強調表示します。
  3. そのテキストの下に表示される **Add Feedback** ポップアップをクリックします。
  4. 表示される手順に従ってください。
- より詳細なフィードバックを行う場合は、Bugzilla の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関するご意見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記入してください。
  4. **Submit Bug** をクリックします。

第1章 OPENSHIFT DATA FOUNDATION の紹介

Red Hat OpenShift Data Foundation は、Red Hat OpenShift Container Platform のクラウドストレージおよびデータサービスの集合です。これは、単純なデプロイメントおよび管理を容易に実行するために Operator としてパッケージ化され、Red Hat OpenShift Container Platform サービスカタログの一部として利用できます。

Red Hat OpenShift Data Foundation サービスは、主に以下のコンポーネントを表すストレージクラスを使用してアプリケーションで使用できます。

- ブロックストレージデバイス。主にデータベースのワークロードに対応します。主な例には、Red Hat OpenShift Container Platform のロギングおよびモニタリング、および PostgreSQL などがあります。
- 共有および分散ファイルシステム。主にソフトウェア開発、メッセージング、およびデータ集約のワークロードに対応します。これらの例には、Jenkins ビルドソースおよびアーティファクト、Wordpress のアップロードコンテンツ、Red Hat OpenShift Container Platform レジストリー、および JBoss AMQ を使用したメッセージングが含まれます。
- Multicloud オブジェクトストレージ。複数のクラウドオブジェクトストアからのデータの保存および取得を抽象化できる軽量 S3 API エンドポイントを特長としています。
- オンプレミスオブジェクトストレージ。主にデータ集約型アプリケーションをターゲットとする数十ペタバイトおよび数十億のオブジェクトにスケーリングする堅牢な S3 API エンドポイントを特長としています。これらの例には、Spark、Presto、Red Hat AMQ Streams (Kafka) などのアプリケーションや、TensorFlow や Pytorch などのマシンラーニングフレームワークを使用した行、列、および半構造化データの保存およびアクセスが含まれます。

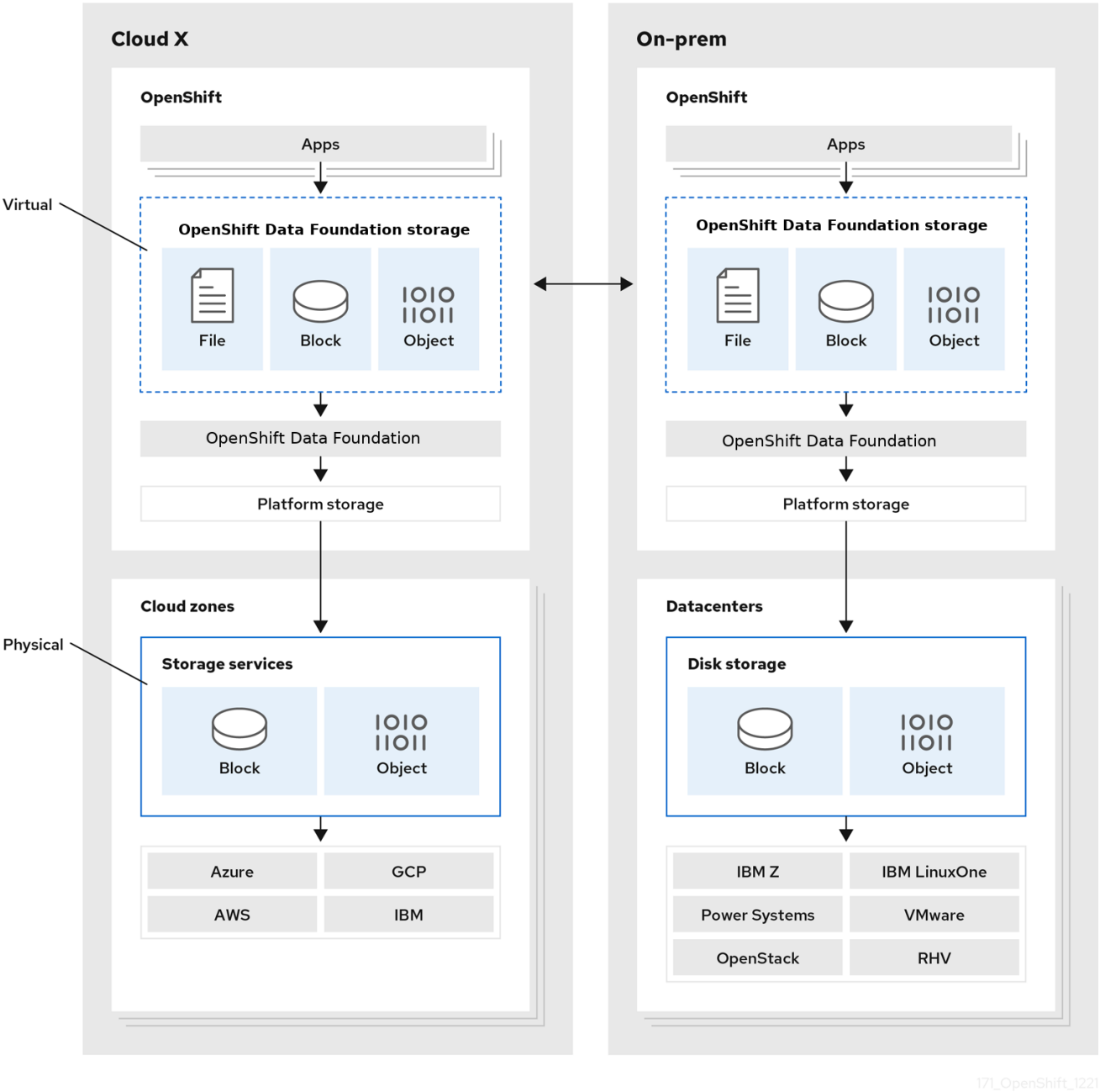
Red Hat OpenShift Data Foundation バージョン 4.x は、以下を含むソフトウェアプロジェクトのコレクションを統合します。

- Ceph。ブロックストレージ、共有および分散ファイルシステム、およびオンプレミスのオブジェクトストレージを提供します。
- Ceph CSI。永続ボリュームおよび要求のプロビジョニングおよびライフサイクルを管理します。
- NooBaa。Multicloud Object Gateway を提供します。
- OpenShift Data Foundation サービスを初期化し、管理する OpenShift Data Foundation、Rook-Ceph、および NooBaa Operator。

第2章 OPENSHIFT DATA FOUNDATION アーキテクチャーの概要

Red Hat OpenShift Data Foundation は、Red Hat OpenShift Container Platform のサービスを提供し、Red Hat OpenShift Container Platform から内部で実行できます。

Red Hat OpenShift Data Foundation アーキテクチャー



Red Hat OpenShift Data Foundation は、インストーラーでプロビジョニングされるインフラストラクチャー、またはユーザーによってプロビジョニングされるインフラストラクチャー でデプロイされる Red Hat OpenShift Container Platform クラスターへのデプロイメントをサポートします。これら 2 つの方法については、OpenShift Container Platform のインストールプロセス について参照してください。Red Hat OpenShift Data Foundation と Red Hat OpenShift Container Platform のコンポーネントの相互運用性についての詳細は、相互運用性マトリックスを参照してください。

OpenShift Container Platform のアーキテクチャーおよびライフサイクルについての詳細は、『OpenShift Container Platform architecture』を参照してください。

第3章 OPENSHIFT DATA FOUNDATION OPERATOR

Red Hat OpenShift Data Foundation は、以下の 3 つの Operator Liftcycle Manager(OLM)Operator バンドルで構成されており、管理タスクとカスタムリソースをコード化し、タスクとリソースの特性を簡単に自動化できるようにします。

- OpenShift Data Foundation
  - **odf-operator**
- OpenShift Container Storage
  - **ocs-operator**
  - **rook-ceph-operator**
- Multicloud Object Gateway
  - **mcg-operator**

管理者はクラスターの必要な最終状態を定義し、OpenShift Data Foundation Operator は管理者の介入を最小限に抑えてクラスターをその状態にするか、またはその状態に近づけるようにします。

3.1. OPENSHIFT DATA FOUNDATION OPERATOR

odf-operatorは、OpenShift Data Foundation の「メタ」オペレーター、つまり、他のオペレーターに影響を与えることを目的としたオペレーターとして説明できます。

odf-operator には、以下の主要機能があります。

- OpenShift Data Foundation を構成する他のオペレーターの設定とバージョン管理を実施します。これは、オペレーターの依存関係とサブスクリプション管理という 2 つの主要なメカニズムを使用して行われます。
  - **odf-operator** バンドルは、他の OLM オペレーターへの依存関係を指定して、それらが常に特定のバージョンでインストールされるようにします。
  - オペレーター自体が他のすべてのオペレーターのサブスクリプションを管理して、それらのオペレーターの目的のバージョンが OLM によるインストールに使用できることを確認します。
- OpenShift Console 用の OpenShift Data Foundation 外部プラグインを提供します。
- ストレージソリューションを OpenShift コンソールと統合するための API を提供します。

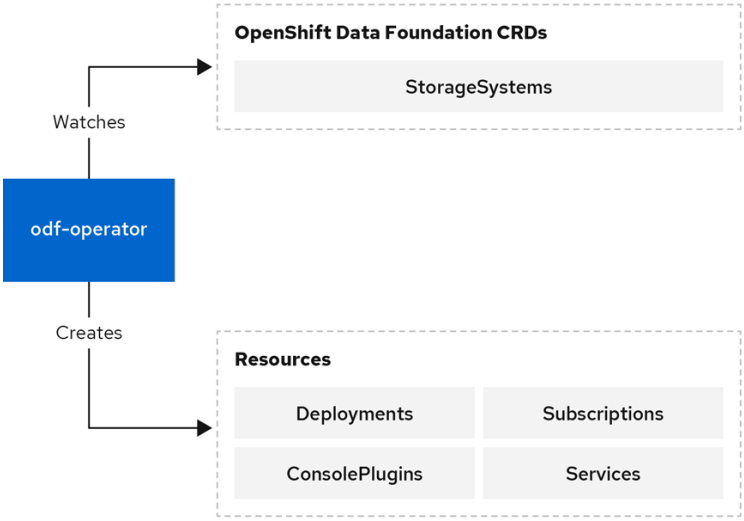
3.1.1. コンポーネント

odf-operatorは、**ocs-operator** パッケージに依存しています。また、**mcg-operator** のサブスクリプションも管理します。さらに、**odf-operator** バンドルは、OpenShift Console 用の OpenShift Data Foundation 外部プラグインの 2 番目のデプロイメントを定義します。これは、OpenShift Data Foundation ダッシュボードを OpenShift Container Platform Console に直接登録および統合するために必要なファイルを提供する **nginx** ベースの Pod を定義します。

3.1.2. 設計ダイアグラム

この図は、**odf-operator** が OpenShift Container Platform にどのように統合されているかを示しています。

図3.1 OpenShift Data Foundation Operator



3.1.3. 責任

odf-operator は、次の CRD を定義します。

- **StorageSystem**

StorageSystem CRD は、OpenShift Container Platform にデータストレージとサービスを提供する基盤となるストレージシステムを表します。これにより、オペレーターは、特定の **種類** のストレージシステムの **サブスクリプション** が存在することを確認します。

3.1.4. リソース

ocs-operator は、特定の StorageSystem の仕様に応じて、次の CR を作成します。

Operator Lifecycle Manager のリソース

指定された **StorageSystem's** の種類を定義および調整するオペレーターの **Subscription** を作成します。

3.1.5. 制限

odf-operator は、データストレージやサービス自体を提供しません。これは、他のストレージシステムの統合および管理レイヤーとして存在します。

3.1.6. 高可用性

高可用性は、他のほとんどの Operator と同様、**odf-operator** Pod の主な要件ではありません。通常、プロセスのディストリビューションに必要な操作や利点はありません。OpenShift Container Platform は、現在の Pod が利用できなくなるか、または削除されるたびに、置換 Pod を迅速に立ち上げます。

3.1.7. 関連する設定ファイル

odf-operator には、演算子の動作を変更するのに使用できる変数の **ConfigMap** が付属しています。

3.1.8. 関連するログファイル

OpenShift Data Foundation の問題の理解を深めるには、以下を確認してください。

- Operator Pod ログ
- **StorageSystem** ステータス
- 基盤となるストレージシステムの CRD ステータス

Operator Pod ログ

各 Operator は、調整および発生したエラーについての情報が含まれる標準の Pod ログを提供します。これらのログには多くの場合、正常な調整に関する情報があり、これを除外して無視することができます。

Storage System のステータスおよびイベント

StorageSystem CR は、CR のステータスに調整の詳細を保存し、関連するイベントを持ちます。**StorageSystem** の仕様には、実際のストレージシステムの CRD の namespace、名前空間、および Kind が含まれています。これらを使用して、管理者はストレージシステムのステータスに関する詳細情報を見つけることができます。

3.1.9. ライフサイクル

OpenShift Data Foundation バンドルがインストールされている限り、**odf-operator** が存在する必要があります。これは OpenShift Data Foundation CSV の OLM の調整の一部として管理されます。Pod の少なくとも 1 つのインスタンスが **Ready** 状態にある必要があります。

CRD などの Operator オペランドは Operator のライフサイクルには影響しません。**StorageSystems** の作成と削除は、オペレーターの制御外の操作であり、管理者が開始するか、適切なアプリケーションプログラミングインターフェイス (API) 呼び出しで自動化する必要があります。

3.2. OPENSHIFT CONTAINER STORAGE OPERATOR

ocs-operator は、OpenShift Data Foundation の「メタ」オペレーター、つまり、他のオペレーターに影響を与えることを目的としたオペレーターであり、他のオペレーターが提供する機能の設定ゲートウェイとして機能します。他の演算子を直接管理しません。

ocs-operator には、以下の主要機能があります。

- 他の Operator をトリガーしてそれらに対して調整するカスタムリソース(CR)を作成します。
- Ceph および Multicloud Object Gateway 設定を抽象化し、それらを Red Hat が検証し、サポートする既知のベストプラクティスに制限します。
- サポートポリシーに従って、コンテナー化された Ceph および NooBaa をデプロイするのに必要なリソースを作成し、調整します。

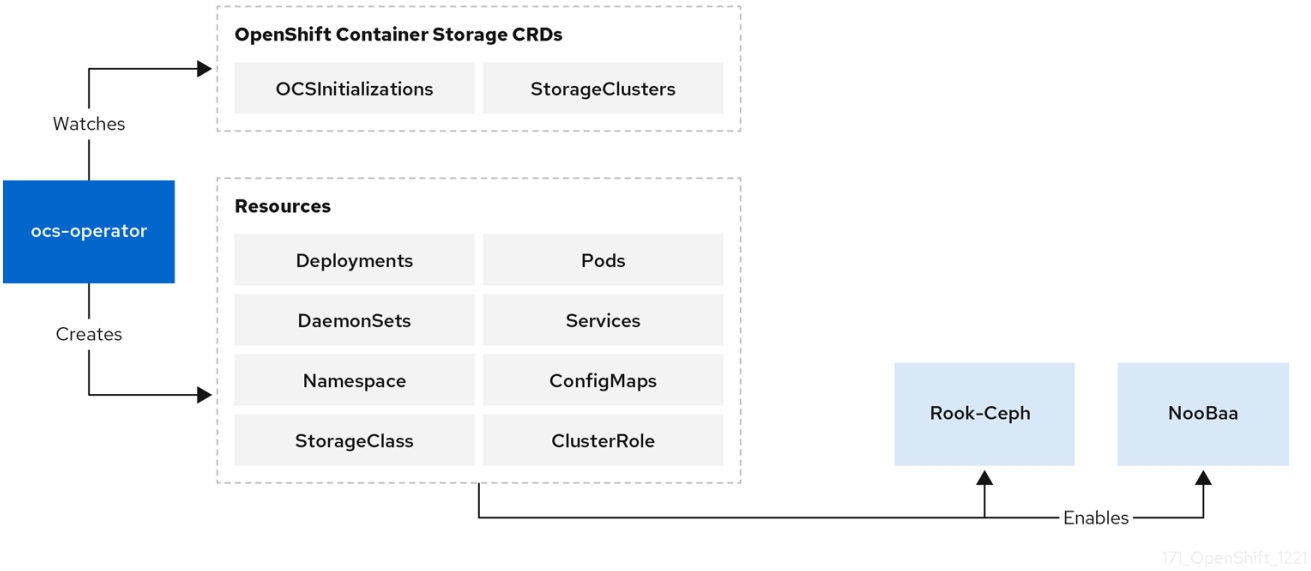
3.2.1. コンポーネント

ocs-operator には依存コンポーネントがありません。ただし、Operator には、**ClusterServiceVersion** (CSV)で定義される他の Operator からすべてのカスタムリソース定義 (CRD)の存在に依存しています。

3.2.2. 設計ダイアグラム

この図は、OpenShift Container Storage が OpenShift Container Platform とどのように統合されているかを示しています。

図3.2 OpenShift Container Storage Operator



3.2.3. 責任

2 つの **ocs-operator** CRD は次のとおりです。

- **OCSInitialization**
- **StorageCluster**

OCSInitialization は、Operator レベルで適用されるカプセル化操作に使用されるシングルトン CRD です。オペレーターは、1つのインスタンスが常に存在することを確認します。CR は以下をトリガーします。

- OpenShift Container Storage に必要な初期化タスクを実行します。必要な場合は、**OCSInitialization** CRD を削除して、これらのタスクを再度実行するようにトリガーすることがされます。
  - OpenShift Container Storage に必要なセキュリティーコンテキスト制約 (SCC) が存在することを確認します。
- 高度なトラブルシューティングおよび復旧操作を実行するために使用される Ceph ツールボックス Pod のデプロイメントを管理します。

StorageCluster CRD は、OpenShift Container Storage の全機能を提供するシステムを表します。これは Operator をトリガーし、CRD **Rook-Ceph** および **NooBaa** の生成および調整を行うことができます。**ocs-operator** アルゴリズムは、**StorageCluster** 仕様の設定に基づいて CRD **CephCluster** および **NooBaa** を生成します。Operator は **CephBlockPools**、**Routes** などの追加の CR も作成します。これらのリソースは、OpenShift Container Storage のさまざまな機能を有効にするのに必要です。現時点で、OpenShift Container Platform クラスターごとに 1 つの StorageCluster CR のみがサポートされます。

3.2.4. リソース

ocs-operatorは、定義するCRDの仕様に応じて、次のCRを作成します。これらのリソースの一部の構成は上書きでき、生成された仕様の変更を許可したり、新規に作成しなくて済むようにできます。

一般的なリソース

イベント

調整に対応するために必要に応じてさまざまなイベントを作成します。

永続ボリューム (PV)

PVはオペレーターが直接作成するものではありません。ただし、Operator は Ceph CSI ドライバーで作成されたすべての PV を追跡し、サポートされる機能に適切なアノテーションが PV にあることを確認します。

クイックスタート

OpenShift Container Platform コンソールのさまざまなクイックスタート CR をデプロイします。

Rook-Ceph リソース

CephBlockPool

デフォルトの Ceph ブロックプールを定義します。Ceph オブジェクトストアの **CephFilesysPrometheusRulesoute**。

StorageClass

デフォルトのストレージクラスを定義します。たとえば、**CephBlockPool** および **CephFilesystem** 用です。

VolumeSnapshotClass

対応するストレージクラスのデフォルトボリュームスナップショットクラスを定義します。

Multicloud Object Gateway リソース

NooBaa

デフォルトの Multicloud Object Gateway システムを定義します。

リソースの監視

- Metrics Exporter Service
- Metrics Exporter Service Monitor
- PrometheusRules

3.2.5. 制限

ocs-operator は OpenShift Data Foundation の他の Pod をデプロイしたり、調整したりしません。**ocs-operator** CSV は、Operator Deployment および Operator Lifecycle Manager (OLM)などの最上位のコンポーネントを定義します。

3.2.6. 高可用性

高可用性は、他のほとんどの Operator と同様、**ocs-operator** Pod の主な要件ではありません。通常、プロセスのディストリビューションに必要な操作や利点はありません。OpenShift Container Platform は、現在の Pod が利用できなくなるか、または削除されるたびに、置換 Pod を迅速に立ち上げます。

3.2.7. 関連する設定ファイル

ocs-operator 設定は CSV によって完全に指定され、CSV のカスタムビルドなければ変更することができません。

3.2.8. 関連するログファイル

OpenShift Container Storage を理解し、問題のトラブルシューティングを行うには、以下を参照してください。

- Operator Pod ログ
- StorageCluster のステータスおよびイベント
- OCSInitialization ステータス

Operator Pod ログ

各 Operator は、調整および発生したエラーについての情報が含まれる標準の Pod ログを提供します。これらのログには多くの場合、正常な調整に関する情報があり、これを除外して無視することができます。

StorageCluster のステータスおよびイベント

StorageCluster CR は、CR のステータスに調整の詳細を保存し、関連するイベントを持ちます。ステータスには、予想されるコンテナーイメージのセクションが含まれます。これは、他の Operator からの Pod に存在する必要があるコンテナーイメージと、現在検出するイメージを示します。これは、OpenShift Container Storage のアップグレードが完了したかどうかを判断するのに役立ちます。

OCSInitialization ステータス

このステータスは、初期化タスクが正常に完了したかどうかを示します。

3.2.9. ライフサイクル

OpenShift Container Storage バンドルがインストールされている限り、**ocs-operator** が存在する必要があります。これは、OLM による OpenShift Container Storage CSV の調整の一部として管理されます。Pod の少なくとも 1 つのインスタンスが **Ready** 状態にある必要があります。

CRD などの Operator オペランドは Operator のライフサイクルには影響しません。**OCSInitialization** CR は常に存在しているはずです。Operator が存在しない場合はこれを作成します。StorageClusters の作成および削除は Operator の制御外の操作であり、管理者によって開始するか、または適切な API 呼び出しによって自動化される必要があります。

3.3. ROOK-CEPH OPERATOR

Rook-Ceph Operator は、OpenShift Data Foundation の Ceph の Rook Operator です。Rookを使用すると、CephストレージシステムをOpenShift Container Platformで実行できます。

Rook-Ceph Operator は、ストレージクラスターを自動的にブートストラップし、ストレージデーモンを監視してストレージクラスターが正常であることを確認する単純なコンテナーです。

3.3.1. コンポーネント

Rook-Ceph Operator は、OpenShift Data Foundation デプロイメントの一部として多数のコンポーネントを管理します。

Ceph-CSI ドライバー

この Operator は、CSI ドライバー（RADOS ブロックデバイス(RBD)および Ceph ファイルシステム(CephFS)のそれぞれに対するプロビジョナーを含む）、ならびにその各ドライバーにボリュームプラグイン **daemonset** を作成し、更新します。

Ceph デーモン

Mons

モニター(mons)は Ceph のコアメタデータストアを提供します。

OSD

オブジェクトストレージデーモン(OSD)は、データを基礎となるデバイスに保存します。

Mgr

マネージャー(mgr)はメトリクスを収集し、Ceph の他の内部機能を提供します。

RGW

RADOS Gateway(RGW)は、オブジェクトストアに S3 エンドポイントを提供します。

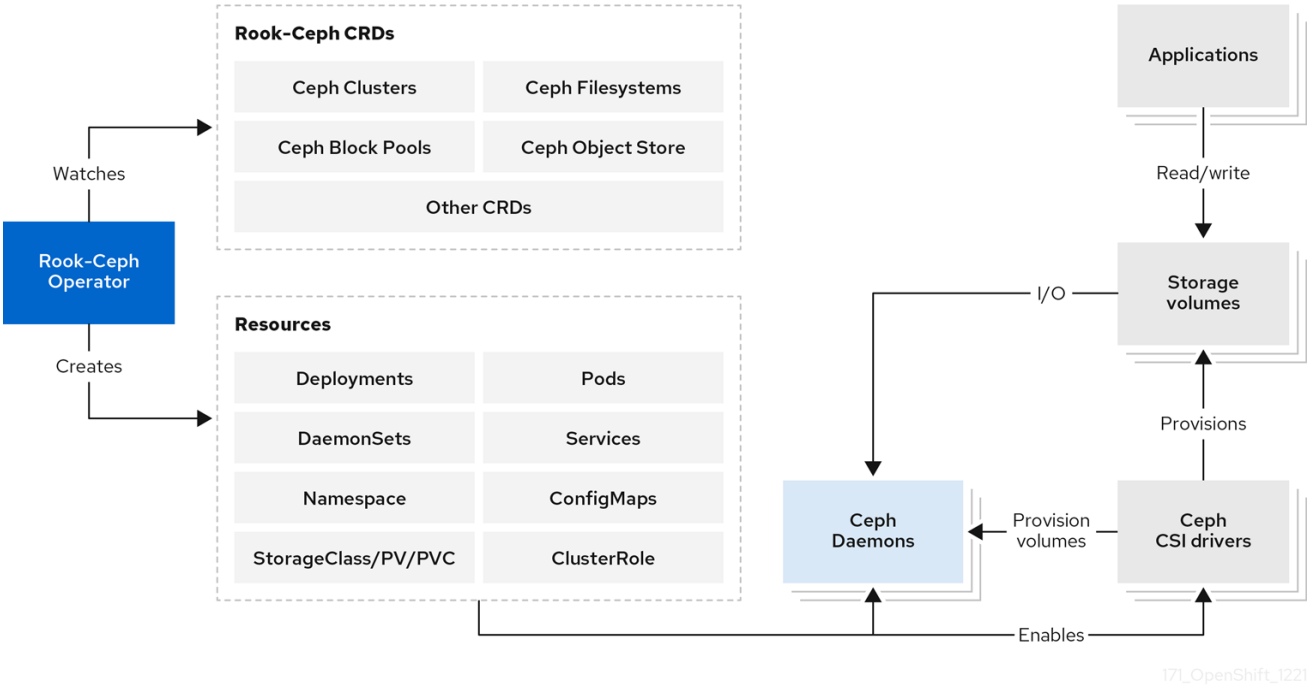
MDS

メタデータサーバー(MDS)は CephFS 共有ボリュームを提供します。

3.3.2. 設計ダイアグラム

以下の図は、Ceph Rook を OpenShift Container Platform と統合する方法を示しています。

図3.3 Rook-Ceph Operator



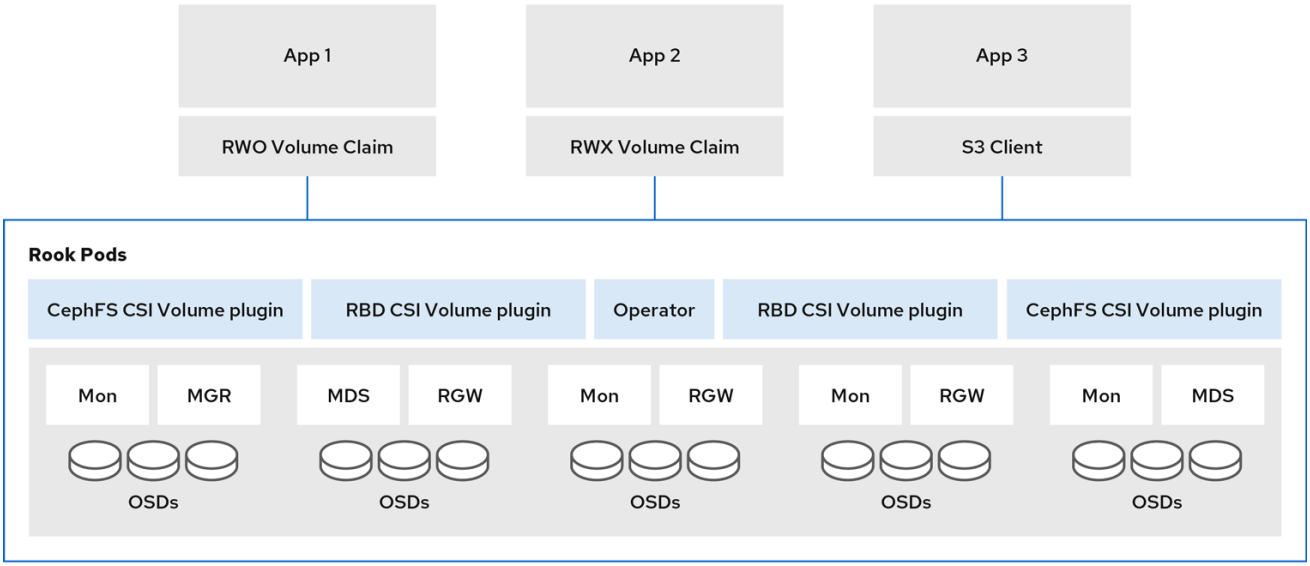
OpenShift Container Platform クラスターで Ceph が実行されている場合、OpenShift Container Platform アプリケーションは、Rook-Ceph によって管理されるブロックデバイスおよびファイルシステムをマウントしたり、オブジェクトストレージに S3/Swift API を使用できます。

3.3.3. 責任

Rook-Ceph Operator は、ストレージクラスターをブートストラップし、監視するコンテナーです。以下の機能を実行します。

- ストレージコンポーネントの設定を自動化する
- Ceph モニター Pod および Ceph OSD デーモンの開始、監視、および管理を行い、RADOS ストレージクラスターを提供する
- Pod およびその他のアーティファクトを初期化し、以下を管理するサービスを実行する
  - プールの CRD
  - オブジェクトストア(S3/Swift)
  - ファイルシステム
- Ceph mons および OSD を監視して、ストレージが利用可能で、正常な状態にあることを確認する
- クラスターサイズに基づいて mon 設定を調整しつつ、Ceph mons 配置をデプロイし、管理する
- API サービスによって要求される必要な状態変更を監視し、変更を適用する
- ストレージの使用に必要な Ceph-CSI ドライバーを初期化する
- ストレージを Pod にマウントするように Ceph-CSI ドライバーを自動的に設定する

Rook-Ceph Operator アーキテクチャー



Rook-Ceph Operator イメージには、クラスターを管理するのに必要なすべてのツールが含まれます。データパスへの変更はありません。ただし、Operator はすべての Ceph 設定を公開しません。配置グループやクラッシュマップなどの Ceph 機能の多くはユーザーに表示されなくなり、物理リソース、プール、ボリューム、ファイルシステム、およびバケットにおけるユーザーエクスペリエンスが改善されました。

3.3.4. リソース

Rook-Ceph Operator は、**openshift-storage** namespace に作成するすべてのリソースに所有者参照を追加します。クラスターがアンインストールされると、所有者参照により、リソースがすべてクリーンアップされるようにします。これには、**configmaps**、**secrets**、**services**、**deployments**,

daemonsets などの OpenShift Container Platform リソースが含まれます。

Rook-Ceph Operator は CR を監視
し、**CephCluster**、**CephObjectStore**、**CephFilesystem**、**CephBlockPool** などの OpenShift Data Foundation によって決定されます。

3.3.5. ライフサイクル

Rook-Ceph Operator は、Ceph クラスターで以下の Pod のライフサイクルを管理します。

Rook Operator

クラスターの調整を所有する単一 Pod。

RBD CSI ドライバー

- 単一デプロイメントで管理される 2 つのプロビジョナー Pod。
- **daemonset** によって管理されるノードごとに 1 つのプラグイン Pod。

CephFS CSI Driver

- 単一デプロイメントで管理される 2 つのプロビジョナー Pod。
- **daemonset** によって管理されるノードごとに 1 つのプラグイン Pod。

モニター (mons)

3 つの mon Pod (それぞれに独自のデプロイメント)。

ストレッチクラスター

5 つの mon Pod (Arbiter ゾーンに 1 つとその他の 2 つのデータゾーンにそれぞれ 2 つずつ) が含まれます。

マネージャー (mgr)

クラスターに単一の mgr Pod があります。

ストレッチクラスター

（OpenShift Data Foundation 4.8 以降では）2 つの mgr Pod があります。1 つは、arbiter 以外の 2 つゾーンにそれぞれ 1 つずつあります。

オブジェクトストレージデーモン (OSD)

3 つ以上の OSD がクラスターに最初に作成されます。クラスターが拡張すると、さらにOSDが追加されます。

メタデータサーバー(MDS)

CephFS メタデータサーバーには単一の Pod があります。

RADOS ゲートウェイ (RGW)

Ceph RGW デーモンには単一の Pod があります。

3.4. MCG OPERATOR

Multicloud Object Gateway (MCG) オペレーターは、OpenShift Data Foundation オペレーターおよび Rook-Ceph オペレーターとともに、OpenShift Data Foundation のオペレーターです。MCG オペレーターは、スタンドアロンオペレーターとしてアップストリームで使用できます。

MCG オペレーターは、次の主要な機能を実行します。

- OpenShift Data Foundation 内で Multicloud Object Gateway (MCG)コンポーネントを制御し、調整します。
- Object Bucket Claim（オブジェクトバケット要求、バケットクラス、バッキングストアなどの新規ユーザーリソース）を管理します。
- デフォルトですぐに使用可能なリソースを作成します。

一部の設定および情報は、OpenShift Data Foundation Operator 経由で MCG Operator に渡されます。

3.4.1. コンポーネント

MCG オペレーターには、サブコンポーネントがありません。ただし、これは、それによって制御されるさまざまなリソースの調整ループで構成されます。

MCG Operator にはコマンドラインインターフェース(CLI)があり、OpenShift Data Foundation の一部として利用できる。これにより、各種リソースの作成、削除、およびクエリーが可能になります。この CLI は、YAML ファイルを直接適用するのではなく、設定が適用される前に、入力サニタイテーションおよびステータス検証のレイヤーを追加します。

3.4.2. 責任およびリソース

MCG Operator はカスタムリソース定義 (CRD) および OpenShift Container Platform エンティティーについて調整されます。

- バッキングストア
- namespace ストア
- Bucket クラス
- Object Bucket Claim（オブジェクトバケット要求）
- NooBaa、Pod ステートフルセット CRD
- Prometheus ルールおよびサービスモニタリング
- Horizontal pod autoscaler (HPA)

バッキングストア

お客様が MCG コンポーネントに接続されているリソース。このリソースは、プロビジョニングされたバケットのデータをMCGに保存する機能を提供します。

デフォルトのバッキングストアは、OpenShift Container Platform が実行しているプラットフォームに応じてデプロイメントの一部として作成されます。たとえば、OpenShift Container Platform または OpenShift Data Foundation が Amazon Web Services (AWS)にデプロイされる場合は、AWS::S3 バケットであるデフォルトのバッキングストアになります。同様に、Microsoft Azure の場合、デフォルトのバッキングストアは Blob コンテナーなどです。

デフォルトのバッキングストアは、OpenShift Container Platform に含まれるクラウド認証情報 Operator の CRD を使用して作成されます。MCG に追加できるバッキングストアの量に制限はありません。バッキングストアは、バケットの異なるポリシーを定義するためにバケットクラス CRD で使用されます。バッキングストアとしてサポートされるサービスまたはリソースのタイプを特定するには、特定の OpenShift Data Foundation バージョンのドキュメントを参照してください。

namespace ストア

namespace バケットで使用されるリソース。デプロイメント時に、デフォルトは作成されません。

Bucketclass

新しくプロビジョニングされたバケットのデフォルトまたは初期ポリシー。以下のポリシーは、バケットクラスに設定されます。

配置ポリシー

バケットに割り当てられるバッキングストアを示し、バケットのデータの書き込みに使用します。このポリシーはデータバケットに使用され、キャッシュポリシーでローカルキャッシュの配置を指定します。配置ポリシーのモードは 2 つあります。

- Spread。定義されたバッキングストア全体でデータを削除します。
- Mirror。各バッキングストアで完全なレプリカを作成します。

Namespace ポリシー

集約に使用されるリソースと、書き込みターゲットに使用されるリソースを定義する namespace バケットのポリシー。

キャッシュポリシー

これはバケットのポリシーで、ハブ（信頼できるソース）と、キャッシュアイテムの存続時間（TTL）を設定します。

デフォルトのバケットクラスはデプロイメント中に作成され、デフォルトのバッキングストアを使用する配置ポリシーで設定されます。追加できるバケットクラスの数に制限はありません。

サポート対象のポリシーの種類を特定するには、特定の OpenShift Data Foundation バージョンのドキュメントを参照してください。

Object Bucket Claim（オブジェクトバケット要求）

S3 バケットのプロビジョニングを有効にする CRD。MCG では、OBC は任意のバケットクラスを受け取り、バケットの初期設定を書き留めておきます。バケットクラスが指定されていない場合は、デフォルトのバケットクラスが使用されます。

NooBaa、Pod ステートフルセット CRD

DB Pod、コア Pod、エンドポイントなどの NooBaa デプロイメントの異なる Pod を制御する内部 CRD。この CRD は内部であるため、変更しないでください。この Operator は以下のエンティティーを調整します。

- DB Pod SCC
- OpenShift Container Platform と NooBaa ユーザーインターフェースとの間の SSO シングルサインオンを許可するロールバインディングおよびサービスアカウント
- S3 アクセスのルート
- OpenShift Container Platform によって取得および署名され、S3 ルートに設定されている証明書

Prometheus ルールおよびサービスのモニタリング

これらのCRDは、MCGでサポートされているPrometheusおよびアラートルールのスクレイピングポイントを設定します。

Horizontal pod autoscaler (HPA)

これは MCG エンドポイントと統合されます。エンドポイントポッドは、CPUの負荷（S3トラフィックの量）に応じてスケールアップおよびスケールダウンします。

3.4.3. 高可用性

Operatorとして提供される唯一の高可用性は、OpenShift Container Platformが障害のあるPodを再スケジュールすることです。

3.4.4. 関連するログファイル

NooBaa Operator の問題のトラブルシューティングを行うには、以下を確認します。

- Operator Pod ログ（must-gather でも利用できます）
- must-gather で利用できる各種 CRD またはエンティティー、およびそのステータス。

3.4.5. ライフサイクル

MCG オペレーターは、OpenShift Data Foundation がデプロイされた後、実行および調整が行われてからアンインストールされます。

第4章 OPENSHIFT DATA FOUNDATION のインストールの概要

OpenShift Data Foundation は、複数のオペレーターによって管理される複数のコンポーネントで構成されています。

4.1. INSTALLED OPERATORS

オペレーターハブから OpenShift Data Foundation をインストールすると、以下の 4 つの個別のデプロイメントが作成されます。

- **odf-operator** : odf-operator Pod を定義します。
- **ocs-operator**: 同じコンテナーで **ocs-operator** およびその **metrics-exporter** のプロセスを実行する **ocs-operator** Pod を定義します。
- **rook-ceph-operator**: **rook-ceph-operator** Pod を定義します。
- MCG-operator: **mcg-operator** Pod を定義します。

これらのオペレーターは独立して実行され、他のオペレーターが監視する顧客リソース (CR) を作成することによって相互に作用します。**ocs-operator** は、主に Ceph ストレージと Multicloud Object Gateway を構成するための CR の作成を担当します。**mcg-operator** は、そのコンポーネントで使用する Ceph ボリュームを作成することがあります。

4.2. OPENSHIFT CONTAINER STORAGE の初期化

OpenShift Data Foundation バンドルは、OpenShift Container Platform Console への外部プラグインも定義し、コンソールでは利用できない新しい画面と機能を追加します。このプラグインは、インストール時に OLM によって作成されたデプロイメントによって管理される **odf-console-plugin** Pod で Web サーバーとして実行されます。

ocs-operator は、作成後に **OCSInitialization** CR を自動的に作成します。どの時点でも、**OCSInitialization** CR は 1 つだけ存在します。これは、単一の **Storage Cluster** のスコープに制限されない **ocs-operator** の動作を制御しますが、それらを実行するのは 1 回だけです。**OCSInitialization** CR を削除すると、**ocs-operator** がそれを再作成します。これにより、初期化操作を再トリガーできます。

OCSInitialization CR は、次の動作を制御します。

SecurityContextConstraints (SCCs)

OCSInitialization CR が作成された後、 **ocs-operator** はコンポーネント Pod で使用するためのさまざまな SCC を作成します。

Ceph Toolbox のデプロイ

OCSInitialization を使用して、高度な Ceph 操作用の Ceph Toolbox Pod をデプロイできます。

Rook-Ceph Operator 設定

この構成は、**rook-ceph-operator** 動作のための全体設定を支配する **rook-ceph-operator-config ConfigMap** を作成します。

4.3. ストレージクラスターの作成

OpenShift Data Foundation オペレーター自体はストレージ機能を提供しないため、必要なストレージ構成を定義する必要があります。

オペレーターをインストールした後、OpenShift Container Platform コンソールウィザードまたは CLI のいずれかを使用して、新しい **StorageCluster** を作成し、**ocs-operator** がこの **StorageCluster** を調整します。OpenShift Data Foundation は、インストールごとに 1 つの **StorageCluster** をサポートします。最初の CR の後に作成された **StorageCluster** CR は、**ocs-operator** によって無視されます。

OpenShift Data Foundation では、以下の 3 つの StorageCluster 構成が可能です。

内部

内部モードでは、すべてのコンポーネントが OpenShift Container Platform クラスター内でコンテナー化されて実行され、インストールウィザードで管理者が指定した **StorageClass** に対して作成された、動的にプロビジョニングされた永続ボリューム (PV) を使用します。

内部割り当て

このモードは内部モードに似ていますが、管理者は、Ceph がバッキングストレージに使用するクラスターノードに直接接続されているローカルストレージデバイスを定義する必要があります。また、管理者は、ローカルストレージオペレーターが **StorageClass** を提供するために調整する CR を作成する必要があります。**ocs-operator** は、この **StorageClass** を Ceph のバッキングストレージとして使用します。

外部

このモードでは、Ceph コンポーネントは OpenShift Container Platform クラスター内で実行されません。代わりに、アプリケーションが PV を作成できる外部の OpenShift Container Storage インストールへの接続が提供されます。他のコンポーネントは、必要に応じてクラスター内で実行されます。

MCG Standalone: このモードは、CephCluster を使用せずに Multicloud Object Gateway システムのインストールを容易にします。

StorageCluster CR が見つかった後、**ocs-operator** はそれを検証し、ストレージコンポーネントを定義するための後続のリソースの作成を開始します。

4.3.1. 内部モードストレージクラスター

内部ストレージクラスターと内部接続ストレージクラスターの両方で、次のような同じセットアッププロセスがあります。

StorageClasses	クラスターアプリケーションが Ceph ボリュームの作成に使用するストレージクラスを作成します。
SnapshotClasses	クラスターアプリケーションが Ceph ボリュームのスナップショットを作成するのに使用するボリュームスナップショットクラスを作成します。
Ceph RGW 設定	さまざまな Ceph オブジェクト CR を作成して、Ceph RGW オブジェクトストレージエンドポイントへのアクセスを有効にして提供します。
Ceph RBD 設定	**CephBlockPool** CR を作成して、RBD ストレージを有効にします。
CephFS 設定	**CephFilesystem** CR を作成して、Ceph FS ストレージを有効にします。
Rook-Ceph 設定	基礎となる Ceph クラスターの全体的な動作を管理する **rook-config-override ConfigMap** を作成します。

CephCluster	**Ceph Cluster** CR を作成して、**rook-ceph-operator** から Ceph 調整をトリガーします。詳細については、Rook-Ceph オペレーター を参照してください。
NoobaaSystem	**NooBaa** CR を作成して、**mcg-operator** からの調整をトリガーします。詳細については、MCG オペレーター を参照してください。
ジョブテンプレート	OpenShift Container Storage の管理操作を実行するジョブを定義する**OpenShift テンプレート** CR を作成します。
クイックスタート	Web コンソールにクイックスタートガイドを表示する **QuickStart** CR を作成します。

4.3.1.1. クラスターの作成

ocs-operator が **CephCluster** CR を作成した後、 **rook-operator** は目的の設定に従って Ceph クラスターを作成します。**rook-operator** は、次のコンポーネントを構成します。

Ceph **mon** デーモン	3 つの Ceph **mon** デーモンは、クラスター内の異なるノード上で開始しています。これらは Ceph クラスターのコアメタデータを管理し、過半数のクォーラムを形成する必要があります。各 **mon** のメタデータは、クラウド環境にある場合は PV によって、ローカルストレージデバイス環境にある場合はローカルホスト上のパスによってサポートされます。
Ceph **mgr** デーモン	このデーモンが起動し、クラスターのメトリックを収集して Prometheus に報告します。
Ceph OSD	これらの OSD は、**storageClassDeviceSets** の構成に従って作成されます。各 OSD は、ユーザーデータを格納する PV を消費します。デフォルトでは、Ceph は、**CRUSH** アルゴリズムを使用して高い耐久性と可用性を実現するために、異なる OSD 間でアプリケーションデータの 3 つのレプリカを維持します。
CSI プロビジョナー	これらのプロビジョナーは RBD と **CephFS** のために開始されます。OpenShift Container Storage のストレージクラスに対してボリュームがリクエストされると、リクエストは **Ceph-CSI** ドライバーに送信され、Ceph でボリュームをプロビジョニングします。
CSI ボリュームプラグインおよび **CephFS**	RBD および **CephFS**の CSI ボリュームプラグインは、クラスター内の各ノードで開始されます。ボリュームプラグインは、Ceph ボリュームをアプリケーションでマウントする必要がある場所で実行する必要があります。

CephCluster CR が構成された後、Rook は残りの Ceph CR を調整してセットアップを完了します。

CephBlockPool	**CephBlockPool** CR は、Rook オペレーターが RWO ボリューム用の Ceph プールを作成するための設定を提供します。
CephFilesystem	**CephFilesystem** CR は、通常 RWX ボリューム用に CephFS を使用して共有ファイルシステムを構成するように Rook オペレーターに指示します。共有ボリュームを管理するために、CephFS メタデータサーバー (MDS) が起動します。

CephObjectStore	**CephObjectStore** CR は、RGW サービスを使用してオブジェクトストアを構成するように Rook オペレーターに指示します。
CephObjectStoreUser CR	**CephObjectStoreUser** CR は、Rook オペレーターに、Noo Baa が消費するオブジェクトストアユーザーを構成し、**access/private** キーと **CephObjectStore** エンドポイントを公開するように指示します。

オペレーターは Ceph の状態を監視して、ストレージプラットフォームが正常な状態を維持していることを確認します。**mon** デーモンが長時間 (10 分) ダウンした場合、Rook はその場所で新しい **mon** を開始し、クォーラム全体を完全に復元できるようにします。

ocs-operator が **CephCluster** CR を更新すると、Rook は要求された変更に即座に応答して、クラスター構成を更新します。

4.3.1.2. NooBaa システムの作成

NooBaa システムが作成されると、**mcg-operator** は以下を調整します。

デフォルトの BackingStore

OpenShift Container Platform および OpenShift Data Foundation がデプロイされているプラットフォームに応じて、バケットが配置ポリシーに使用できるように、デフォルトのバッキングストアリソースが作成されます。さまざまなオプションは次のとおりです。

Amazon Web Services (AWS) のデプロイ	**mcg-operator** は、**CloudCredentialsOperator** (CCO) を使用して認証情報を作成し、新しい AWS::S3 バケットを作成し、そのバケットの上に **BackingStore** を作成します。
Microsoft Azure のデプロイ	**mcg-operator** は、CCO を使用して認証情報を作成し、新しい Azure Blob を作成し、そのバケットの上に **BackingStore** を作成します。
Google Cloud Platform (GCP) のデプロイ	**mcg-operator** は、CCO を使用して認証情報を作成して新しい GCP バケットを作成し、そのバケットに BackingStore を作成します。
オンプレミスデプロイメント	RGW が存在する場合、**mcg-operator** は RGW の上に新しい**CephUser** と新しいバケットを作成し、そのバケットの上に **BackingStore** を作成します。
前述のデプロイメントはいずれも適用できません	**mcg-operator** は、デフォルトのストレージクラスに基づいて **pv-pool** を作成し、そのバケットの上に **BackingStore** を作成します。

デフォルトの BucketClass

デフォルトの **BucketClass** への配置ポリシーを持つ **BackingStore**。

NooBaa Pod

次の Noo Baa Pod が作成され、開始します。

データベース (DB)	これは、メタデータ、統計、イベントなどを保持する Postgres DB です。ただし、保存されている実際のデータは保持されません。
コア	これは、構成、バックグラウンドプロセス、メタデータ管理、統計などを処理する Pod です。
エンドポイント	これらの Pod は、重複排除や圧縮、さまざまなサービスとの通信によるデータの書き込みと読み取りなど、実際の I/O 関連の作業を実行します。エンドポイントは、**HorizonalPodAutoscaler** と統合されており、既存のエンドポイント Pod で観察された CPU 使用率に応じて、エンドポイントの数が増減します。

ルート

NooBaa S3 インターフェイスのルートは、S3 を使用するアプリケーション用に作成されます。

サービス

NooBaa S3 インターフェイスのサービスは、S3 を使用するアプリケーション用に作成されます。

4.3.2. 外部モードストレージクラスター

外部ストレージクラスターの場合、**ocs-operator** はわずかに異なるセットアッププロセスに従います。**ocs-operator** は、**rook-ceph-external-cluster-details ConfigMap** の存在を探しますが、これは管理者かコンソールのどちらか他の人が作成したものでなければなりません。**ConfigMap** の作成方法については、Creating an OpenShift Data Foundation Cluster for external mode を参照してください。次に、**ocs-operator** は、**ConfigMap** で指定されているように、次のリソースの一部またはすべてを作成します。

外部 Ceph 設定	外部 **mons** のエンドポイントを指定する ConfigMap。
外部 Ceph 認証情報シークレット	外部 Ceph インスタンスに接続するための認証情報を含むシークレット。
外部 Ceph StorageClasses	RBD、Ceph FS、および/または RGW のボリュームの作成を可能にする 1 つ以上の StorageClasses。
Ceph FSCSI ドライバーを有効にする	**CephFS** StorageClass が指定されている場合は、**rook-ceph-operator** は、**Ceph FS**CSI Pod をデプロイします。
Ceph RGW 設定	RGW StorageClass が指定されている場合は、さまざまな Ceph Object CR を作成して、Ceph RGW オブジェクトストレージエンドポイントへのアクセスを有効にして提供します。

ConfigMap で指定されたリソースを作成した後、StorageCluster の作成プロセスは次のように進行します。

CephCluster	**Ceph Cluster** CR を作成して、**rook-ceph-operator** から Ceph 調整をトリガーします (後続のセクションを参照)。
SnapshotClasses	アプリケーションが Ceph ボリュームのスナップショットを作成するために使用する **SnapshotClasses** を作成します。
NoobaaSystem	**Noo Baa** CR を作成して、noobaa-operator からの調整をトリガーします (後続のセクションを参照)。

Quickstarts : コンソールの クイック スタートガイドを表示するクイックスタート CR を作成します。

4.3.2.1. クラスターの作成

CephCluster CR が外部モードで作成されると、Rook オペレーターは次の操作を実行します。

- オペレーターは、リモート Ceph クラスターへの接続が使用可能であることを検証します。接続には、**mon** エンドポイントとシークレットをローカルクラスターにインポートする必要があります。
- CSI ドライバーは、Ceph へのリモート接続で構成されます。RBD および **CephFS** プロビジョナーとボリュームプラグインは、内部モードで設定された場合、CSI ドライバーと同様に開始され、Ceph への接続は OpenShift クラスターの外部にあります。
- モニターアドレスの変更を定期的に監視し、それに応じて Ceph-CSI 設定を更新します。

4.3.2.2. NooBaa システムの作成

NooBaa システムが作成されると、**mcg-operator** は以下を調整します。

デフォルトの BackingStore

OpenShift Container Platform および OpenShift Data Foundation がデプロイされているプラットフォームに応じて、バケットが配置ポリシーに使用できるように、デフォルトのバッキングストアリソースが作成されます。さまざまなオプションは次のとおりです。

Amazon Web Services (AWS) のデプロイ	**mcg-operator** は、**CloudCredentialsOperator** (CCO) を使用して認証情報を作成し、新しい AWS::S3 バケットを作成し、そのバケットの上に **BackingStore** を作成します。
Microsoft Azure のデプロイ	**mcg-operator** は、CCO を使用して認証情報を作成し、新しい Azure Blob を作成し、そのバケットの上に **BackingStore** を作成します。
Google Cloud Platform (GCP) のデプロイ	**mcg-operator** は、CCO を使用して認証情報を作成して新しい GCP バケットを作成し、そのバケットに BackingStore を作成します。

オンプレミスデプロイメント	RGW が存在する場合、**mcg-operator** は RGW の上に新しい**CephUser** と新しいバケットを作成し、そのバケットの上に **BackingStore** を作成します。
前述のデプロイメントはいずれも適用できません	**mcg-operator** は、デフォルトのストレージクラスに基づいて **pv-pool** を作成し、そのバケットの上に **BackingStore** を作成します。

デフォルトの BucketClass

デフォルトの **BucketClass** への配置ポリシーを持つ **BackingStore**。

NooBaa Pod

次の Noo Baa Pod が作成され、開始します。

データベース (DB)	これは、メタデータ、統計、イベントなどを保持する Postgres DB です。ただし、保存されている実際のデータは保持されません。
コア	これは、構成、バックグラウンドプロセス、メタデータ管理、統計などを処理する Pod です。
エンドポイント	これらの Pod は、重複排除や圧縮、さまざまなサービスとの通信によるデータの書き込みと読み取りなど、実際の I/O 関連の作業を実行します。エンドポイントは、**HorizonalPodAutoscaler** と統合されており、既存のエンドポイント Pod で観察された CPU 使用率に応じて、エンドポイントの数が増減します。

ルート

NooBaa S3 インターフェイスのルートは、S3 を使用するアプリケーション用に作成されます。

サービス

NooBaa S3 インターフェイスのサービスは、S3 を使用するアプリケーション用に作成されます。

4.3.3. MCG Standalone StorageCluster

このモードでは、CephCluster は作成されません。代わりに、デフォルト値を使用して NooBaa システム CR が作成され、OpenShift Container Platform ダッシュボードの既存の StorageClasses を利用します。

4.3.3.1. NooBaa システムの作成

NooBaa システムが作成されると、**mcg-operator** は以下を調整します。

デフォルトの BackingStore

OpenShift Container Platform および OpenShift Data Foundation がデプロイされているプラットフォームに応じて、バケットが配置ポリシーに使用できるように、デフォルトのバッキングストアリソースが作成されます。さまざまなオプションは次のとおりです。

Amazon Web Services (AWS) のデプロイ	**mcg-operator** は、**CloudCredentialsOperator** (CCO) を使用して認証情報を作成し、新しい AWS::S3 バケットを作成し、そのバケットの上に **BackingStore** を作成します。
Microsoft Azure のデプロイ	**mcg-operator** は、CCO を使用して認証情報を作成し、新しい Azure Blob を作成し、そのバケットの上に **BackingStore** を作成します。
Google Cloud Platform (GCP) のデプロイ	**mcg-operator** は、CCO を使用して認証情報を作成して新しい GCP バケットを作成し、そのバケットに BackingStore を作成します。
オンプレミスデプロイメント	RGW が存在する場合、**mcg-operator** は RGW の上に新しい**CephUser** と新しいバケットを作成し、そのバケットの上に **BackingStore** を作成します。
前述のデプロイメントはいずれも適用できません	**mcg-operator** は、デフォルトのストレージクラスに基づいて **pv-pool** を作成し、そのバケットの上に **BackingStore** を作成します。

デフォルトの BucketClass

デフォルトの **BucketClass** への配置ポリシーを持つ **BackingStore**。

NooBaa Pod

次の Noo Baa Pod が作成され、開始します。

データベース (DB)	これは、メタデータ、統計、イベントなどを保持する Postgres DB です。ただし、保存されている実際のデータは保持されません。
コア	これは、構成、バックグラウンドプロセス、メタデータ管理、統計などを処理する Pod です。
エンドポイント	これらの Pod は、重複排除や圧縮、さまざまなサービスとの通信によるデータの書き込みと読み取りなど、実際の I/O 関連の作業を実行します。エンドポイントは、**HorizonalPodAutoscaler** と統合されており、既存のエンドポイント Pod で観察された CPU 使用率に応じて、エンドポイントの数が増減します。

ルート

NooBaa S3 インターフェイスのルートは、S3 を使用するアプリケーション用に作成されます。

サービス

NooBaa S3 インターフェイスのサービスは、S3 を使用するアプリケーション用に作成されます。

4.3.3.2. StorageSystem Creation

StorageCluster 作成の一部として、**odf-operator** は対応する **StorageSystem** CR を自動的に作成します。これにより、StorageCluster が OpenShift Data Foundation に公開されます。

第5章 OPENSHIFT DATA FOUNDATION アップグレードの概要

OpenShift Data Foundation は、Operator Lifecycle Manager (OLM) によって管理されるオペレーターバンドルとして、**ClusterServiceVersion** (CSV) CR によって製品のインストールやアップグレードなどのハイレベルなタスクを行うためにオペレーターを活用しています。

5.1. ワークフローのアップグレード

OpenShift Data Foundation は、Z-stream リリースアップグレードとマイナーバージョンのリリースアップグレードの 2 種類のアップグレードを認識します。この 2 つのアップグレードパスのユーザーインターフェイスのワークフローは完全に同じではありませんが、その動作はかなり似ています。違いは次のとおりです。

Z-ストリームのリリースでは、OCS は **redhat-operators CatalogSource** で新しいバンドルを公開します。OLM はこれを検出し、既存の CSV を置き換える新しい CSV の **InstallPlan** を作成します。サブスクリプション承認戦略は、自動か手動かにかかわらず、OLM が調整を続行するか、管理者の承認を待つかを決定します。

マイナーバージョンのリリースでは、OpenShift Container Storage も **redhat-operators CatalogSource** で新しいバンドルを公開します。違いは、このバンドルが新しいチャンネルの一部になるということで、チャンネルのアップグレードは自動的には行われません。管理者は、新しいリリースチャンネルを明示的に選択する必要があります。これが完了すると、OLM はこれを検出し、既存の CSV を置き換える新しい CSV の**Install Plan** を作成します。チャンネルスイッチは手動操作であるため、OLM は自動的に調整を開始します。

この時点以降、アップグレードプロセスは同じです。

5.2. CLUSTERSERVICEVERSION 調整

OLM は、承認された **InstallPlan** を検出すると、CSV を調整するプロセスを開始します。大まかに言えば、これは、新しい仕様に基づいてオペレーターリソースを更新し、新しい CSV が正しくインストールされていることを確認してから、古い CSV を削除することによって行われます。アップグレードプロセスにより、オペレーターのデプロイメントに更新がプッシュされ、新しい CSV で指定されたイメージを使用してオペレーター Pod の再起動がトリガーされます。

注記

特定の CSV に変更を加えて、それらの変更を関連するリソースに反映させることは可能ですが、新しい CSV にアップグレードすると、変更されていない仕様に基づいて新しい CSV が作成されるため、変更した内容はすべて失われます。

5.3. オペレーターの調整

この時点で、OpenShift Data Foundation オペランドの調整は、OpenShift Data Foundation installation overview のインストールの概要 で定義されているとおりに進行します。オペレーターは、関連するすべてのリソースが、ユーザー向けのリソース (たとえば、**StorageCluster**) で指定されている予想される設定で存在することを確認します。