Red Hat Ceph Storage 7

オペレーションガイド

Red Hat Ceph Storage の操作タスク

Last Updated: 2024-03-09

Red Hat Ceph Storage 7 オペレーションガイド

Red Hat Ceph Storage の操作タスク

法律上の通知

概要

本書では、Red Hat Ceph Storage で動作するタスクを実行する方法について説明します。 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 を参照してください。

目次

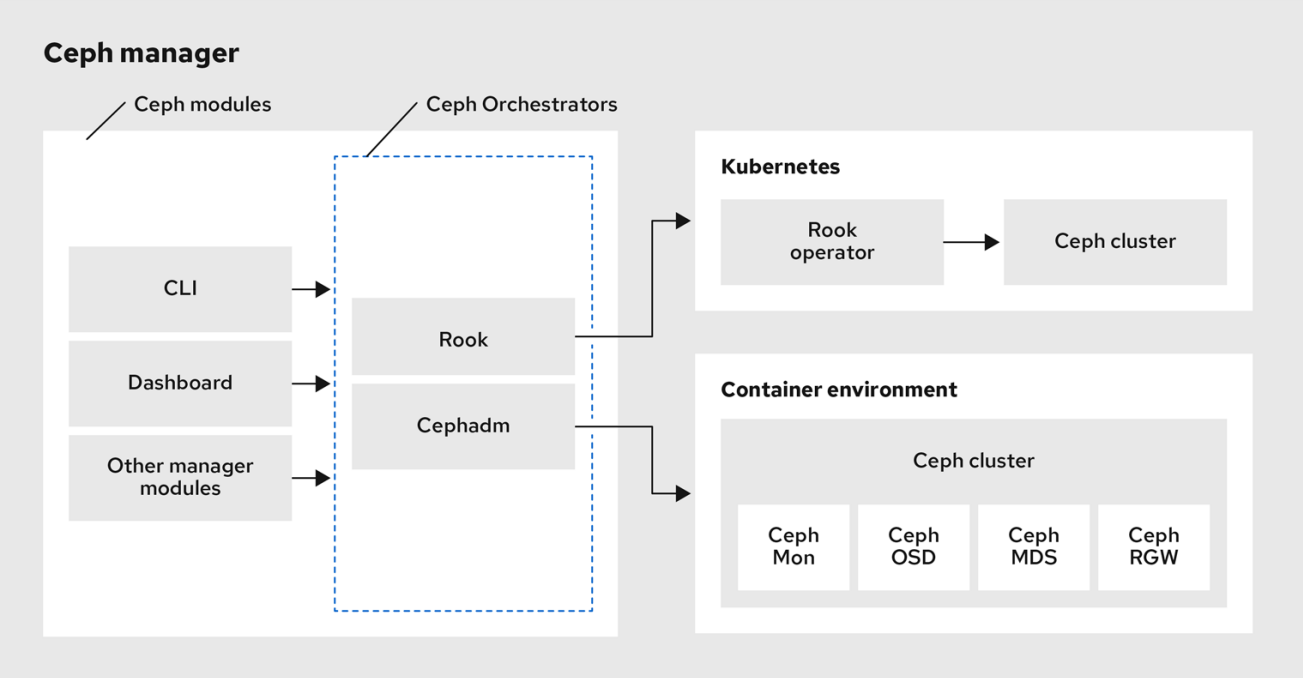
第1章 CEPH ORCHESTRATOR の紹介

ストレージ管理者として、Red Hat Ceph Storage クラスターでデバイスを検出し、サービスを作成する機能を提供する Ceph Orchestrator with Cephadm ユーティリティーを使用できます。

1.1. CEPH ORCHESTRATOR の使用

Red Hat Ceph Storage Orchestrators は、主に Red Hat Ceph Storage クラスターと、Rook や Cephadm などのデプロイメントツール間のブリッジとして機能するマネージャーモジュールです。また、Ceph コマンドラインインターフェイスおよび Ceph Dashboard と統合します。

以下は、Ceph Orchestrator のワークフローの図です。



153_Ceph_0421

Red Hat Ceph Storage Orchestrators の種類

Red Hat Ceph Storage Orchestrators には、主に 3 つのタイプがあります。

- **Orchestrator CLI**: これらは Orchestrators で使用される一般的な API で、実装できるコマンドセットが含まれます。これらの API は、**ceph-mgr** モジュールを外部オーケストレーションサービスとオーケストレートする一般的なコマンドラインインターフェイス (CLI) も提供しています。以下は、Ceph Orchestrator で使用される用語です。
  - ホスト: これは、物理ホストのホスト名で、コンテナー内の Pod 名、DNS 名、コンテナー名、またはホスト名ではありません。
  - サービスタイプ: これは、nfs、mds、osd、mon、rgw、mgr などのサービスのタイプです。
  - サービス: モニターサービス、マネージャーサービス、OSD サービス、Ceph Object Gateway サービス、NFS サービスなど、Ceph Storage クラスターが提供する機能サービス。

- デーモン: Ceph Object Gateway サービスなどの 1 つ以上のホストによってデプロイされるサービスの特定のインスタンスでは、3 つの異なるホストで異なる Ceph Object Gateway デーモンを実行することができます。

- **Cephadm Orchestrator**: これは、Rook または Ansible などの外部ツールに依存しない Ceph Orchestrator モジュールであり、SSH 接続を確立し、明示的な管理コマンドを実行することでクラスターのノードを管理します。このモジュールは Day One および Day Two 操作を対象としています。
  Cephadm Orchestrator の使用は、Ansible などのデプロイメントフレームワークを使用せずに Ceph Storage クラスターをインストールする推奨方法です。これは、ストレージデバイスのインベントリーの作成、OSD のデプロイメントと交換、Ceph デーモンの起動と停止など、あらゆる管理操作を実行するために、クラスター内のすべてのノードに接続できる SSH 設定および鍵にアクセスできるマネージャーデーモンを提供するものです。さらに、Cephadm Orchestrator は、同一の場所に配置されたサービスの独立したアップグレードを可能にするために、**systemd** によって管理されるコンテナーイメージをデプロイします。

  また、このオーケストレーターでは、Ceph Monitor および Ceph Manager を実行する最低限のクラスターをブートストラップするコマンドなど、現在のホストでコンテナーイメージベースのサービスのデプロイメントを管理するために、必要なすべての操作をカプセル化するツールが重要なポイントとなるでしょう。

- **Rook Orchestrator**: Rook は、Kubernetes Rook operator を使用して Kubernetes クラスター内で実行される Ceph ストレージクラスターを管理するオーケストレーションツールです。rook モジュールは、Ceph の Orchestrator フレームワークと Rook との間の統合を提供します。
  Rook は、Kubernetes のオープンソースクラウドネイティブストレージ operator です。Rook は operator モデルに従い、ここでは Ceph ストレージクラスターおよびその望ましい状態を記述するためにカスタムリソース定義 (CRD) オブジェクトが Kubernetes で定義されます。rook operator デーモンは、現在のクラスター状態と望ましい状態を比較する制御ループで実行され、それらを収束させるための手順を実行します。Ceph の望ましい状態を記述する主なオブジェクトは Ceph ストレージクラスター CRD で、どのデバイスが OSD によって消費されるか、実行すべきモニターの数、使用すべき Ceph バージョンなどの情報が含まれます。
  Rook は、RBD プール、CephFS ファイルシステムなどを記述するために他の複数の CRD を定義します。

  Rook Orchestrator モジュールは、**ceph-mgr** デーモンで実行されるグルーで、必要なクラスターの状態を記述する Kubernetes で Ceph ストレージクラスターに変更を加えて Ceph オーケストレーション API を実装します。Rook クラスターの **ceph-mgr** デーモンは Kubernetes Pod として実行されているため、rook モジュールは明示的な設定がなくても Kubernetes API に接続できます。

第2章 CEPH ORCHESTRATOR を使用したサービスの管理

ストレージ管理者は、Red Hat Ceph Storage クラスターのインストール後に、Ceph Orchestrator を使用してストレージクラスターのサービスを監視および管理できます。サービスは、一緒に設定されるデーモンのグループです。

本セクションでは、以下の管理情報について説明します。

- Ceph Orchestrator の配置仕様
- コマンドラインインターフェイスを使用した Ceph デーモンのデプロイ
- コマンドラインインターフェイスを使用したホストのサブセットでの Ceph デーモンのデプロイ
- Ceph Orchestrator のサービス仕様
- サービス仕様を使用した Ceph デーモンのデプロイ
- サービス仕様を使用した Ceph File System ミラーリングデーモンのデプロイ

2.1. CEPH ORCHESTRATOR の配置仕様

Ceph Orchestrator を使用して、**osds**、**mons**、**mgrs**、**mds**、および **rgw** サービスをデプロイできます。Red Hat は、配置仕様を使用してサービスをデプロイすることを推奨します。Ceph Orchestrator を使用して、サービスをデプロイするためにデプロイする必要があるデーモンの場所と数を把握する必要があります。配置仕様は、コマンドライン引数または **yaml** ファイルのサービス仕様として渡すことができます。

配置仕様を使用してサービスをデプロイする方法は 2 つあります。

- コマンドラインインターフェイスで配置仕様を直接使用します。たとえば、ホストに 3 つのモニターをデプロイする場合は、以下のコマンドを実行して **host01**、**host02**、および **host03** に 3 つのモニターをデプロイします。

  例

  ```
  [ceph: root@host01 /]# ceph orch apply mon --placement="3 host01 host02 host03"
  ```

- YAML ファイルでの配置仕様の使用。たとえば、すべてのホストに **node-exporter** をデプロイする場合には、**yaml** ファイルで以下を指定できます。

  例

  ```
  service_type: node-exporter
  placement:
  host_pattern: '*'
  extra_entrypoint_args:
    - "--collector.textfile.directory=/var/lib/node_exporter/textfile_collector2"
  ```

2.2. コマンドラインインターフェイスを使用した CEPH デーモンのデプロイ

Ceph Orchestrator を使用すると、**ceph orch** コマンドを使用して Ceph Manager、Ceph Monitors、Ceph OSD、モニタリングスタックなどのデーモンをデプロイできます。配置仕様は、Orchestrator コマンドの **--placement** 引数として渡されます。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- ホストがストレージクラスターに追加されている。

手順

1. Cephadm シェルにログインします。

   例

   ```
   [root@host01 ~]# cephadm shell
   ```

2. 以下のいずれかの方法を使用して、ホストにデーモンをデプロイします。

   - **方法 1**: デーモンの数とホスト名を指定します。

     構文

     ```
     ceph orch apply SERVICE_NAME --placement="NUMBER_OF_DAEMONS HOST_NAME_1 HOST_NAME_2 HOST_NAME_3"
     ```

     例

     ```
     [ceph: root@host01 /]# ceph orch apply mon --placement="3 host01 host02 host03"
     ```

   - **方法 2**: ラベルをホストに追加してから、ラベルを使用してデーモンをデプロイします。

     a. ホストにラベルを追加します。

        構文

        ```
        ceph orch host label add HOSTNAME_1 LABEL
        ```

        例

        ```
        [ceph: root@host01 /]# ceph orch host label add host01 mon
        ```

     b. ラベルでデーモンをデプロイします。

        構文

        ```
        ceph orch apply DAEMON_NAME label:LABEL
        ```

        例

        ```
        ceph orch apply mon label:mon
        ```

- **方法 3**: ラベルをホストに追加し、**--placement** 引数を使用してデプロイします。

  a. ホストにラベルを追加します。

  構文

  ```
  ceph orch host label add HOSTNAME_1 LABEL
  ```

  例

  ```
  [ceph: root@host01 /]# ceph orch host label add host01 mon
  ```

  b. ラベルの配置仕様を使用してデーモンをデプロイします。

  構文

  ```
  ceph orch apply DAEMON_NAME --placement="label:LABEL"
  ```

  例

  ```
  ceph orch apply mon --placement="label:mon"
  ```

検証

- サービスをリスト表示します。

  例

  ```
  [ceph: root@host01 /]# ceph orch ls
  ```

- ホスト、デーモン、およびプロセスをリスト表示します。

  構文

  ```
  ceph orch ps --daemon_type=DAEMON_NAME
  ceph orch ps --service_name=SERVICE_NAME
  ```

  例

  ```
  [ceph: root@host01 /]# ceph orch ps --daemon_type=mon
  [ceph: root@host01 /]# ceph orch ps --service_name=mon
  ```

関連情報

- **Red Hat Ceph Storage オペレーションガイド**の Ceph Orchestrator を使用したホストの追加 セクションを参照してください。

2.3. コマンドラインインターフェイスを使用したホストのサブセットへの CEPH デーモンのデプロイ

--placement オプションを使用して、ホストのサブセットにデーモンをデプロイできます。デーモンをデプロイするホストの名前で、配置仕様のデーモン数を指定できます。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- ホストがクラスターに追加されている。

手順

1. Cephadm シェルにログインします。

   例

   ```
   [root@host01 ~]# cephadm shell
   ```

2. Ceph デーモンをデプロイするホストをリスト表示します。

   例

   ```
   [ceph: root@host01 /]# ceph orch host ls
   ```

3. デーモンをデプロイします。

   構文

   ```
   ceph orch apply SERVICE_NAME --placement="NUMBER_OF_DAEMONS
   HOST_NAME_1 _HOST_NAME_2 HOST_NAME_3"
   ```

   例

   ```
   ceph orch apply mgr --placement="2 host01 host02 host03"
   ```

   この例では、**mgr** デーモンは、2 つのホストにのみデプロイされます。

検証

- ホストをリスト表示します。

  例

  ```
  [ceph: root@host01 /]# ceph orch host ls
  ```

関連情報

- Red Hat Ceph Storage Operations Guideの Ceph Orchestrator を使用したホストの一覧表示 セクションを参照してください。

2.4. CEPH ORCHESTRATOR のサービス仕様

サービス仕様は、Ceph サービスのデプロイに使用されるサービス属性および設定を指定するデータ構造です。以下は、サービス仕様を指定するためのマルチドキュメント YAML ファイル **cluster.yml** の例になります。

例

```
service_type: mon
placement:
  host_pattern: "mon*"
---
service_type: mgr
placement:
  host_pattern: "mgr*"
---
service_type: osd
service_id: default_drive_group
placement:
  host_pattern: "osd*"
data_devices:
  all: true
```

以下は、サービス仕様のプロパティーが以下のように定義されるパラメーターのリストです。

- **service_type**: サービスのタイプ
  - mon、crash、mds、mgr、osd、rbd、rbd-mirror などの Ceph サービス。
  - nfs や rgw などの Ceph ゲートウェイ。
  - Alertmanager、Prometheus、Grafana、または Node-exporter などのモニタリングスタック。
  - カスタムコンテナーのコンテナー。
- **service_id**: サービスの一意名
- **placement**: これは、デーモンの場所とデプロイ方法を定義するために使用されます。
- **unmanaged**: **true** に設定すると、Orchestrator は、このサービスに関連付けられたデーモンをデプロイまたは削除しません。

Orchestrator のステートレスサービス

ステートレスサービスは、状態の情報を利用可能にする必要がないサービスです。たとえば、**rgw** サービスを起動するには、サービスの起動または実行に追加情報は必要ありません。**rgw** サービスは、機能を提供するためにこの状態に関する情報を作成しません。**rgw** サービスを開始されるタイミングに関係なく、状態は同じになります。

2.5. サービス仕様を使用した CEPH デーモンのデプロイ

Ceph Orchestrator を使用すると、YAML ファイルのサービス仕様を使用して Ceph Manager、Ceph Monitors、Ceph OSD、モニタリングスタックなどのデーモンをデプロイできます。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。

- すべてのノードへの root レベルのアクセス。

手順

1. **yaml** ファイルを作成します。

   例

   ```
   [root@host01 ~]# touch mon.yaml
   ```

2. このファイルは 2 つの方法で設定できます。

   - ファイルを編集して、配置仕様にホストの詳細を含めます。

     構文

     ```
     service_type: SERVICE_NAME
     placement:
       hosts:
         - HOST_NAME_1
         - HOST_NAME_2
     ```

     例

     ```
     service_type: mon
     placement:
       hosts:
         - host01
         - host02
         - host03
     ```

   - ファイルを編集し、ラベルの詳細を配置仕様に含めます。

     構文

     ```
     service_type: SERVICE_NAME
     placement:
       label: "LABEL_1"
     ```

     例

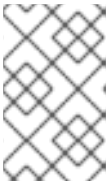
     ```
     service_type: mon
     placement:
       label: "mon"
     ```

3. オプション: サービスのデプロイ中に、CPU、CA 証明書、その他のファイルなどのサービス仕様ファイルで追加のコンテナー引数を使用することもできます。

   例

   ```
   extra_container_args:
     - "-v"
     - "/etc/pki/ca-trust/extracted:/etc/pki/ca-trust/extracted:ro"
   ```

```
    - "--security-opt"
    - "label=disable"
    - "cpus=2"
    - "--collector.textfile.directory=/var/lib/node_exporter/textfile_collector2"
```



注記

Red Hat Ceph Storage 6.1 以降のリリースでは、Cephadm によってデプロイされたノードエクスポーターで追加のメトリックを有効にするための追加の引数の使用がサポートされています。

4. YAML ファイルをコンテナー内のディレクトリーにマウントします。

   例

   ```
   [root@host01 ~]# cephadm shell --mount mon.yaml:/var/lib/ceph/mon/mon.yaml
   ```

5. そのディレクトリーに移動します。

   例

   ```
   [ceph: root@host01 /]# cd /var/lib/ceph/mon/
   ```

6. サービス仕様を使用して Ceph デーモンをデプロイします。

   構文

   ```
   ceph orch apply -i FILE_NAME.yaml
   ```

   例

   ```
   [ceph: root@host01 mon]# ceph orch apply -i mon.yaml
   ```

検証

- サービスをリスト表示します。

  例

  ```
  [ceph: root@host01 /]# ceph orch ls
  ```

- ホスト、デーモン、およびプロセスをリスト表示します。

  構文

  ```
  ceph orch ps --daemon_type=DAEMON_NAME
  ```

  例

  ```
  [ceph: root@host01 /]# ceph orch ps --daemon_type=mon
  ```

関連情報

- Red Hat Ceph Storage Operations Guideの Ceph Orchestrator を使用したホストの一覧表示 セクションを参照してください。

2.6. サービス仕様を使用した CEPH FILE SYSTEM ミラーリングデーモンのデプロイ

Ceph File System (CephFS) は、CephFS ミラーリングデーモン (**cephfs-mirror**) を使用した、リモートの CephFS ファイルシステムへのスナップショットの非同期レプリケーションをサポートします。スナップショットの同期は、スナップショットデータをリモート CephFS にコピーし、同じ名前のリモートターゲットに新しいスナップショットを作成します。Ceph Orchestrator を使用すると、YAML ファイルのサービス仕様を使用して **cephfs-mirror** をデプロイできます。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- すべてのノードへの root レベルのアクセス。
- CephFS が作成されている。

手順

1. **yaml** ファイルを作成します。

   例

   ```
   [root@host01 ~]# touch mirror.yaml
   ```

2. ファイルを編集して以下を追加します。

   構文

   ```
   service_type: cephfs-mirror
   service_name: SERVICE_NAME
   placement:
     hosts:
       - HOST_NAME_1
       - HOST_NAME_2
       - HOST_NAME_3
   ```

   例

   ```
   service_type: cephfs-mirror
   service_name: cephfs-mirror
   placement:
     hosts:
       - host01
       - host02
       - host03
   ```

3. YAML ファイルをコンテナー内のディレクトリーにマウントします。

例

```
[root@host01 ~]# cephadm shell --mount mirror.yaml:/var/lib/ceph/mirror.yaml
```

4. そのディレクトリーに移動します。

   例

   ```
   [ceph: root@host01 /]# cd /var/lib/ceph/
   ```

5. サービス仕様を使用して **cephfs-mirror** デーモンをデプロイします。

   例

   ```
   [ceph: root@host01 /]# ceph orch apply -i mirror.yaml
   ```

検証

- サービスをリスト表示します。

  例

  ```
  [ceph: root@host01 /]# ceph orch ls
  ```

- ホスト、デーモン、およびプロセスをリスト表示します。

  例

  ```
  [ceph: root@host01 /]# ceph orch ps --daemon_type=cephfs-mirror
  ```

関連情報

- **cephfs-mirror** デーモンの詳細は、Ceph File System のミラー を参照してください。

第3章 CEPH ORCHESTRATOR を使用したホストの管理

ストレージ管理者は、バックエンドで Cephadm で Ceph Orchestrator を使用し、既存の Red Hat Ceph Storage クラスターでホストを追加、リスト表示、および削除できます。

ホストにラベルを追加することもできます。ラベルは自由形式であり、特別な意味はありません。各ホストに複数のラベルを指定できます。たとえば、Monitor デーモンがデプロイされたすべてのホストに **mon** ラベルを適用し、Manager デーモンがデプロイされたすべてのホストに **mgr** を適用し、Ceph オブジェクトゲートウェイに **rgw** を適用します。

ストレージクラスター内のすべてのホストにラベルを付けると、各ホストで実行されているデーモンをすばやく識別できるため、システム管理タスクが簡素化されます。さらに、Ceph Orchestrator または YAML ファイルを使用して、特定のホストラベルを持つホストにデーモンをデプロイまたは削除できます。

本セクションでは、以下の管理タスクを説明します。

- **Ceph Orchestrator を使用したホストの追加。**
- **Ceph Orchestrator を使用した複数ホストの追加。**
- **Ceph Orchestrator を使用したホストのリスト表示。**
- **Ceph Orchestrator を使用したホストへのラベルの追加。**
- **Ceph Orchestrator を使用したホストの削除。**
- **Ceph Orchestrator を使用したホストのメンテナンスモード。**

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- すべてのノードへの root レベルのアクセス。
- 新しいホストの IP アドレスは **/etc/hosts** ファイルで更新する必要があります。

3.1. CEPH ORCHESTRATOR を使用したホストの追加

バックエンドで Cephadm で Ceph Orchestrator を使用して、ホストを既存の Red Hat Ceph Storage クラスターに追加できます。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- ストレージクラスター内のすべてのノードへの root レベルのアクセス。
- ノードを CDN に登録して、サブスクリプションを割り当てます。
- ストレージクラスター内のすべてのノードへの sudo アクセスおよびパスワードなしの **ssh** アクセスのある Ansible ユーザー。

手順

1. Ceph 管理ノードから、Cephadm シェルにログインします。

例

```
[root@host01 ~]# cephadm shell
```

2. クラスターの SSH 公開鍵をフォルダーにデプロイメントします。

構文

```
ceph cephadm get-pub-key > ~/PATH
```

例

```
[ceph: root@host01 /]# ceph cephadm get-pub-key > ~/ceph.pub
```

3. Ceph クラスターの SSH 公開鍵を、新たなホストの root ユーザーの **authorized_keys** ファイルにコピーします。

構文

```
ssh-copy-id -f -i ~/PATH root@HOST_NAME_2
```

例

```
[ceph: root@host01 /]# ssh-copy-id -f -i ~/ceph.pub root@host02
```

4. Ansible 管理ノードから、新しいホストを Ansible インベントリーファイルに追加します。ファイルのデフォルトの場所は **/usr/share/cephadm-ansible/hosts/** です。以下の例は、一般的なインベントリーファイルの構造を示しています。

例

```
host01
host02
host03

[admin]
host00
```

注記

以前に新しいホストを Ansible インベントリーファイルに追加し、ホストでプリフライト Playbook を実行している場合は、ステップ 6 に進みます。

5. **--limit** オプションを指定して、プリフライト Playbook を実行します。

構文

```
ansible-playbook -i INVENTORY_FILE cephadm-preflight.yml --extra-vars
"ceph_origin=rhcs" --limit NEWHOST
```

例

```
[ceph-admin@admin cephadm-ansible]$ ansible-playbook -i hosts cephadm-preflight.yml --extra-vars "ceph_origin=rhcs" --limit host02
```

プリフライト Playbook は、新しいホストに **podman**、**lvm2**、**chronyd**、および **cephadm** をインストールします。インストールが完了すると、**cephadm** は **/usr/sbin/** ディレクトリーに配置されます。

6. Ceph 管理ノードから、Cephadm シェルにログインします。

   例

   ```
   [root@host01 ~]# cephadm shell
   ```

7. **cephadm** オーケストレーターを使用して、ストレージクラスターにホストを追加します。

   構文

   ```
   ceph orch host add HOST_NAME IP_ADDRESS_OF_HOST [--label=LABEL_NAME_1,LABEL_NAME_2]
   ```

   --label オプションは任意です。これを使用すると、ホストの追加時にラベルが追加されます。ホストには複数のラベルを追加できます。

   例

   ```
   [ceph: root@host01 /]# ceph orch host add host02 10.10.128.70 --labels=mon,mgr
   ```

検証

- ホストをリスト表示します。

  例

  ```
  [ceph: root@host01 /]# ceph orch host ls
  ```

関連情報

- Red Hat Ceph Storage Operations Guideの Ceph Orchestrator を使用したホストの一覧表示 セクションを参照してください。
- **cephadm-preflight** Playbook の詳細は、**Red Hat Ceph Storage インストールガイド** の プリフライト playbook の実行 セクションを参照してください。
- **Red Hat Ceph Storage インストールガイド** の Red Hat Ceph Storage ノードの CDN への登録およびサブスクリプションの割り当て セクションを参照してください。
- **Red Hat Ceph Storage インストールガイド** の sudo アクセスのある Ansible ユーザーの作成 セクションを参照してください。

3.2. CEPH ORCHESTRATOR を使用した複数ホストの追加

YAML ファイル形式のサービス仕様の使用と同時に、Ceph Orchestrator を使用して複数のホストを Red Hat Ceph Storage クラスターに追加することができます。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。

手順

1. **hosts.yaml** ファイルを作成します。

   例

   ```
   [root@host01 ~]# touch hosts.yaml
   ```

2. **hosts.yaml** ファイルを編集し、以下の詳細を含めます。

   例

   ```
   service_type: host
   addr: host01
   hostname: host01
   labels:
   - mon
   - osd
   - mgr
   ---
   service_type: host
   addr: host02
   hostname: host02
   labels:
   - mon
   - osd
   - mgr
   ---
   service_type: host
   addr: host03
   hostname: host03
   labels:
   - mon
   - osd
   ```

3. YAML ファイルをコンテナー内のディレクトリーにマウントします。

   例

   ```
   [root@host01 ~]# cephadm shell --mount hosts.yaml:/var/lib/ceph/hosts.yaml
   ```

4. そのディレクトリーに移動します。

   例

   ```
   [ceph: root@host01 /]# cd /var/lib/ceph/
   ```

5. サービス仕様を使用してホストをデプロイします。

構文

```
ceph orch apply -i FILE_NAME.yaml
```

例

```
[ceph: root@host01 hosts]# ceph orch apply -i hosts.yaml
```

検証

- ホストをリスト表示します。

例

```
[ceph: root@host01 /]# ceph orch host ls
```

関連情報

- Red Hat Ceph Storage Operations Guideの Ceph Orchestrator を使用したホストの一覧表示 セクションを参照してください。

3.3. CEPH ORCHESTRATOR を使用したホストのリスト表示

Ceph クラスターのホストを Ceph Orchestrator で リスト表示できます。

注記

ホストの **STATUS** は、**ceph orch host ls** コマンドの出力では空白になります。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- ホストがストレージクラスターに追加されている。

手順

1. Cephadm シェルにログインします。

例

```
[root@host01 ~]# cephadm shell
```

2. クラスターのホストをリスト表示します。

例

```
[ceph: root@host01 /]# ceph orch host ls
```

ホストの **STATUS** が空白であることを確認できます。これは想定内です。

3.4. CEPH ORCHESTRATOR を使用したホストへのラベルの追加

Ceph Orchestrator を使用して、既存の Red Hat Ceph Storage クラスター内のホストにラベルを追加できます。ラベルの例の一部は、ホストにデプロイされるサービスに基づいて、**mgr**、**mon**、および **osd** になります。

cephadm に特別な意味を持ち、_ で始まる以下のホストラベルを追加することもできます。

- **_no_schedule**: このラベルは、**cephadm** がホスト上でデーモンをスケジュールまたはデプロイすることを阻止します。すでに Ceph デーモンが含まれている既存のホストに追加されると、これにより、**cephadm** は、自動的に削除されない OSD を除いて、それらのデーモンを別の場所に移動します。ホストに **_no_schedule** ラベルが追加されると、デーモンはそのホストにデプロイされません。ホストが削除される前にデーモンがドレインされると、そのホストに **_no_schedule** ラベルが設定されます。

- **_no_autotune_memory**: このラベルは、ホスト上のメモリーを自動調整しません。そのホスト上の 1 つ以上のデーモンに対して、**osd_memory_target_autotune** オプションまたは他の同様のオプションが有効になっている場合でも、デーモンメモリーが調整されることを阻止します。

- **_admin**: デフォルトでは、**_admin** ラベルはストレージクラスター内のブートストラップされたホストに適用され、**client.admin** キーは、**ceph orch client-keyring {ls|set|rm}** 関数でそのホストに配布されるように設定されます。このラベルを追加のホストに追加すると、通常、**cephadm** は設定ファイルとキーリングファイルを **/etc/ceph** ディレクトリー にデプロイします。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- ホストがストレージクラスターに追加される。

手順

1. Cephadm シェルにログインします。

   例

   ```
   [root@host01 ~]# cephadm shell
   ```

2. ホストにラベルを追加します。

   構文

   ```
   ceph orch host label add HOST_NAME LABEL_NAME
   ```

   例

   ```
   [ceph: root@host01 /]# ceph orch host label add host02 mon
   ```

検証

- ホストをリスト表示します。

  例

  ```
  [ceph: root@host01 /]# ceph orch host ls
  ```

3.5. CEPH ORCHESTRATOR を使用したホストの削除

Ceph Orchestrator で、Ceph クラスターのホストを削除できます。すべてのデーモンは、**_no_schedule** ラベルを追加する **drain** オプションで削除され、操作が完了するまでデーモンまたはクラスターをデプロイメントできないようにします。

重要

ブートストラップホストを削除する場合は、ホストを削除する前に、必ず管理キーリングと設定ファイルをストレージクラスター内の別のホストにコピーしてください。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- すべてのノードへの root レベルのアクセス。
- ホストがストレージクラスターに追加されている。
- すべてのサービスがデプロイされている。
- Cephadm が、サービスを削除する必要があるノードにデプロイされている。

手順

1. Cephadm シェルにログインします。

   例

   ```
   [root@host01 ~]# cephadm shell
   ```

2. ホストの詳細を取得します。

   例

   ```
   [ceph: root@host01 /]# ceph orch host ls
   ```

3. ホストからすべてのデーモンをドレインします。

   構文

   ```
   ceph orch host drain HOSTNAME
   ```

   例

   ```
   [ceph: root@host01 /]# ceph orch host drain host02
   ```

_no_schedule ラベルは、デプロイメントをブロックするホストに自動的に適用されます。

4. OSD の削除のステータスを確認します。

   例

   ```
   [ceph: root@host01 /]# ceph orch osd rm status
   ```

   OSD に配置グループ (PG) が残っていない場合、OSD は廃止され、ストレージクラスターから削除されます。

5. ストレージクラスターからすべてのデーモンが削除されているかどうかを確認します。

   構文

   ```
   ceph orch ps HOSTNAME
   ```

   例

   ```
   [ceph: root@host01 /]# ceph orch ps host02
   ```

6. ホストを削除。

   構文

   ```
   ceph orch host rm HOSTNAME
   ```

   例

   ```
   [ceph: root@host01 /]# ceph orch host rm host02
   ```

関連情報

- 詳細は **Red Hat Ceph Storage オペレーションガイド** の Ceph Orchestrator を使用したホストの追加 セクションを参照してください。
- 詳細は **Red Hat Ceph Storage オペレーションガイド** の Ceph Orchestrator を使用したホストの一覧表示 セクションを参照してください。

3.6. CEPH ORCHESTRATOR を使用したホストのメンテナンスモード

Ceph Orchestrator を使用して、ホストのメンテナンスモードを切り替えできます。**ceph orch host maintenance enter** コマンドは、**systemd target** を停止します。これにより、ホスト上のすべての Ceph デーモンが停止します。同様に、**ceph orch host maintenance exit** コマンドは **systemd target** を再起動し、Ceph デーモンは自動的に再起動します。

ホストがメンテナンス状態になると、オーケストレータは次のワークフローを採用します。

1. **orch host ok-to stop** コマンドを実行して、ホストの削除がデータの可用性に影響しないことを確認します。
2. ホストに Ceph OSD デーモンがある場合、ホストサブツリーに **noout** を適用して、計画されたメンテナンススロット中にデータ移行がトリガーされないようにします。

3. Ceph ターゲットを停止して、すべてのデーモンを停止します。
4. ホストで **ceph target** を無効にして、再起動によって Ceph サービスが自動的に開始されないようにします。

メンテナンスを終了すると、上記の順序が逆になります。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- すべてのノードへの root レベルのアクセス。
- クラスターに追加されたホスト。

手順

1. Cephadm シェルにログインします。

   例

   ```
   [root@host01 ~]# cephadm shell
   ```

2. ホストをメンテナンスモードにしたり、メンテナンスモードから解除したりできます。
   - ホストをメンテナンスモードにします。

     構文

     ```
     ceph orch host maintenance enter HOST_NAME [--force]
     ```

     例

     ```
     [ceph: root@host01 /]# ceph orch host maintenance enter host02 --force
     ```

     --force フラグを使用すると、ユーザーは警告を回避できますが、アラートは回避できません。

   - ホストをメンテナンスモードから解除します。

     構文

     ```
     ceph orch host maintenance exit HOST_NAME
     ```

     例

     ```
     [ceph: root@host01 /]# ceph orch host maintenance exit host02
     ```

検証

- ホストをリスト表示します。

  例

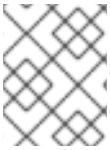
```
[ceph: root@host01 /]# ceph orch host ls
```

第4章 CEPH ORCHESTRATOR を使用したモニターの管理

ストレージ管理者は、配置仕様を使用して追加のモニターをデプロイし、サービス仕様を使用してモニターを追加し、サブネット設定にモニターを追加し、特定のホストにモニターを追加できます。さらに、Ceph Orchestrator を使用してモニターを削除できます。

デフォルトでは、一般的な Red Hat Ceph Storage クラスターには、3 つまたは 5 つのモニターデーモンが異なるホストにデプロイされます。

Red Hat は、クラスターに 5 つ以上のノードがある場合に、5 つのモニターをデプロイすることを推奨します。



注記

Red Hat では、Ceph を OSP director とともにデプロイする場合、3 つのモニターをデプロイすることを推奨します。

Ceph は、クラスターの拡大とともにモニターデーモンを自動的にデプロイし、クラスターの縮小とともにモニターデーモンを自動的にスケールバックします。このような自動拡大および縮小のスムーズな実行は、適切なサブネット設定によります。

モニターノードまたはクラスター全体が単一のサブネットにある場合、Cephadm は新規ホストをクラスターに追加する際に最大 5 つのモニターデーモンを自動的を追加します。Cephadm は、新しいホストでモニターデーモンを自動的に設定します。新しいホストは、ストレージクラスターのブートストラップされたホストと同じサブネットにあります。

Cephadm はストレージクラスターのサイズの変更に対応するようモニターをデプロイし、スケーリングすることもできます。

4.1. CEPH MONITOR

Ceph Monitor は、ストレージクラスターマップのマスターコピーを維持する軽量プロセスです。すべての Ceph クライアントは Ceph モニターに問い合わせ、ストレージクラスターマップの現在のコピーを取得し、クライアントがプールにバインドし、読み取りと書き込みを可能にします。

Ceph Monitor は Paxos プロトコルのバリエーションを使用して、ストレージクラスター全体でマップやその他の重要な情報について合意を確立します。Paxos の性質上、Ceph は、クォーラムを確立するためにモニターの大部分を実行する必要があり、合意を確立します。

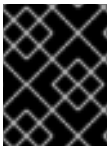
重要

Red Hat では、実稼働クラスターのサポートを受け取るために、別のホストで少なくとも 3 つのモニターが必要になります。

Red Hat は、奇数のモニターをデプロイすることを推奨します。奇数の Ceph Monitors は、偶数のモニターよりも障害に対する回復力が高くなっています。たとえば、2 つのモニターデプロイメントでクォーラムを維持するには、Ceph は障害を許容できません。3 つのモニターでは障害を 1 つ、4 つのモニターでは障害を 1 つ、5 つのモニターでは障害を 2 つ許容します。このため、奇数も推奨されています。要約すると、Ceph は、モニターの大部分 (3 つのうち 2 つ、 4 つのうち 3 つなど) が実行され、相互に通信できるようにする必要があります。

マルチノードの Ceph ストレージクラスターの初回のデプロイには、Red Hat では 3 つのモニターが必要です。3 つ以上のモニターが有効な場合には、一度に数を 2 つ増やします。

Ceph Monitor は軽量であるため、OpenStack ノードと同じホストで実行できます。ただし、Red Hat は、別のホストでモニターを実行することを推奨します。



重要

Red Hat は、コンテナー化された環境における Ceph サービスを共存させることのみをサポートしています。

ストレージクラスターからモニターを削除する場合、Ceph Monitor は Paxos プロトコルを使用して、マスターストレージクラスターマップに関する合意を確立することを検討してください。クォーラムを確立するには、十分な数の Ceph モニターが必要です。

関連情報

- サポートされているすべての Ceph 設定については、ナレッジベースアーティクル Red Hat Ceph Storage でサポートされる設定 を参照してください。

4.2. モニタリング選択ストラテジーの設定

モニター選択ストラテジーは、ネット分割を識別し、障害を処理します。選択モニターストラテジーは、3 つの異なるモードで設定できます。

1. **classic** - これは、2 つのサイト間のエレクターモジュールに基づいて、最も低いランクのモニターが投票されるデフォルトのモードです。

2. **disallow** - このモードでは、モニターを不許可とマークできます。この場合、モニターはクォーラムに参加してクライアントにサービスを提供しますが、選出されたリーダーになることはできません。これにより、許可されていないリーダーのリストにモニターを追加できます。モニターが許可されていないリストにある場合、そのモニターは常に別のモニターに先送りされます。

3. **connectivity** - このモードは、主にネットワークの不一致を解決するために使用されます。各モニターがピアに対して提供する、liveness をチェックする ping に基づいて接続スコアを評価し、最も接続性が高く信頼性の高いモニターをリーダーに選択します。このモードは、クラスターが複数のデータセンターにまたがっている場合や影響を受けやすい場合に発生する可能性のあるネット分割を処理するように設計されています。このモードでは接続スコア評価が組み込まれ、最良スコアのモニターが選択されます。特定のモニターをリーダーにする必要がある場合は、特定のモニターがリスト内でランクが **0** の最初のモニターになるように選択ストラテジーを設定します。

他のモードで機能が必要でない限り、Red Hat は、**classic** モードに留まります。

クラスターを構築する前に、以下のコマンドで **election_strategy** を **classic**、**disallow**、または **connectivity** に変更します。

構文

```
ceph mon set election_strategy {classic|disallow|connectivity}
```

4.3. コマンドラインインターフェイスを使用した CEPH モニターデーモンのデプロイ

Ceph Orchestrator はデフォルトで 1 つのモニターデーモンをデプロイします。コマンドラインイン

ターフェイスで **placement** 仕様を使用して、追加のモニターデーモンをデプロイできます。異なる数のモニターデーモンをデプロイするには、別の数を指定します。モニターデーモンがデプロイされるホストを指定しないと、Ceph Orchestrator はホストをランダムに選択し、モニターデーモンをそれらにデプロイします。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- ホストがクラスターに追加されている。

手順

1. Cephadm シェルにログインします。

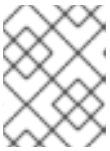
   例

   ```
   [root@host01 ~]# cephadm shell
   ```

2. Ceph モニターデーモンをデプロイするには、以下の 4 つの方法があります。

方法 1

- 配置仕様を使用して、ホストにモニターをデプロイします。



注記

Red Hat は **--placement** オプションを使用して特定のホストにデプロイすることを推奨します。

構文

```
ceph orch apply mon --placement="HOST_NAME_1 HOST_NAME_2 HOST_NAME_3"
```

例

```
[ceph: root@host01 /]# ceph orch apply mon --placement="host01 host02 host03"
```

注記

コマンドの最初のノードとしてブートストラップノードが含まれるようにしてください。

重要

モニターを個別に追加しないでください。**ceph orch apply mon** は置き換えを行うため、モニターはすべてのホストに追加されません。たとえば、以下のコマンドを実行すると、最初のコマンドが **host01** にモニターを作成します。2 番目のコマンドは host1 のモニターの代わりに **host02** にモニターを作成します。3 番目のコマンドは **host02** のモニターの代わりに **host03** にモニターを作成します。最終的には、3 番目のホストにのみモニターが存在することになります。

```
# ceph orch apply mon host01
# ceph orch apply mon host02
# ceph orch apply mon host03
```

方法 2

- 配置仕様を使用して、ラベルの付いた特定ホストに特定数のモニターをデプロイします。

  a. ホストにラベルを追加します。

  構文

  ```
  ceph orch host label add HOSTNAME_1 LABEL
  ```

  例

  ```
  [ceph: root@host01 /]# ceph orch host label add host01 mon
  ```

  b. デーモンをデプロイします。

  構文

  ```
  ceph orch apply mon --placement="HOST_NAME_1:mon HOST_NAME_2:mon
  HOST_NAME_3:mon"
  ```

  例

  ```
  [ceph: root@host01 /]# ceph orch apply mon --placement="host01:mon host02:mon
  host03:mon"
  ```

方法 3

- 配置仕様を使用して、特定ホストに特定数のモニターをデプロイします。

  構文

  ```
  ceph orch apply mon --placement="NUMBER_OF_DAEMONS HOST_NAME_1
  HOST_NAME_2 HOST_NAME_3"
  ```

  例

  ```
  [ceph: root@host01 /]# ceph orch apply mon --placement="3 host01 host02 host03"
  ```

方法 4

- ストレージクラスターのホストにモニターデーモンを無作為にデプロイします。

  構文

  ```
  ceph orch apply mon NUMBER_OF_DAEMONS
  ```

  例

  ```
  [ceph: root@host01 /]# ceph orch apply mon 3
  ```

検証

- サービスをリスト表示します。

  例

  ```
  [ceph: root@host01 /]# ceph orch ls
  ```

- ホスト、デーモン、およびプロセスをリスト表示します。

  構文

  ```
  ceph orch ps --daemon_type=DAEMON_NAME
  ```

  例

  ```
  [ceph: root@host01 /]# ceph orch ps --daemon_type=mon
  ```

4.4. サービス仕様を使用した CEPH モニターデーモンのデプロイ

Ceph Orchestrator はデフォルトで 1 つのモニターデーモンをデプロイします。YAML 形式のファイルなどのサービス仕様を使用して、追加のモニターデーモンをデプロイできます。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- ホストがクラスターに追加されている。

手順

1. **mon.yaml** ファイルを作成します。

   例

   ```
   [root@host01 ~]# touch mon.yaml
   ```

2. **mon.yml** ファイルを編集し、以下の詳細を含めます。

構文

```
service_type: mon
placement:
  hosts:
    - HOST_NAME_1
    - HOST_NAME_2
```

例

```
service_type: mon
placement:
  hosts:
    - host01
    - host02
```

3. YAML ファイルをコンテナー内のディレクトリーにマウントします。

   例

   ```
   [root@host01 ~]# cephadm shell --mount mon.yaml:/var/lib/ceph/mon/mon.yaml
   ```

4. そのディレクトリーに移動します。

   例

   ```
   [ceph: root@host01 /]# cd /var/lib/ceph/mon/
   ```

5. モニターデーモンをデプロイします。

   構文

   ```
   ceph orch apply -i FILE_NAME.yaml
   ```

   例

   ```
   [ceph: root@host01 mon]# ceph orch apply -i mon.yaml
   ```

検証

- サービスをリスト表示します。

  例

  ```
  [ceph: root@host01 /]# ceph orch ls
  ```

- ホスト、デーモン、およびプロセスをリスト表示します。

  構文

  ```
  ceph orch ps --daemon_type=DAEMON_NAME
  ```

例

```
[ceph: root@host01 /]# ceph orch ps --daemon_type=mon
```

4.5. CEPH ORCHESTRATOR を使用した特定ネットワークでのモニターデーモンのデプロイ

Ceph Orchestrator はデフォルトで 1 つのモニターデーモンをデプロイします。各モニターに IP アドレスまたは CIDR ネットワークを明示的に指定し、各モニターが配置される場所を制御できます。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- ホストがクラスターに追加されている。

手順

1. Cephadm シェルにログインします。

   例

   ```
   [root@host01 ~]# cephadm shell
   ```

2. 自動化されたモニターのデプロイメントを無効にします。

   例

   ```
   [ceph: root@host01 /]# ceph orch apply mon --unmanaged
   ```

3. 特定ネットワーク上のホストにモニターをデプロイします。

   構文

   ```
   ceph orch daemon add mon HOST_NAME_1:IP_OR_NETWORK
   ```

   例

   ```
   [ceph: root@host01 /]# ceph orch daemon add mon host03:10.1.2.123
   ```

検証

- サービスをリスト表示します。

  例

  ```
  [ceph: root@host01 /]# ceph orch ls
  ```

- ホスト、デーモン、およびプロセスをリスト表示します。

  構文

```
ceph orch ps --daemon_type=DAEMON_NAME
```

例

```
[ceph: root@host01 /]# ceph orch ps --daemon_type=mon
```

4.6. CEPH ORCHESTRATOR を使用したモニターデーモンの削除

ホストからモニターデーモンを削除するには、他のホストにモニターデーモンを再デプロイします。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- ホストがクラスターに追加されている。
- ホストにデプロイされたモニターデーモン 1 つ以上。

手順

1. Cephadm シェルにログインします。

   例

   ```
   [root@host01 ~]# cephadm shell
   ```

2. **ceph orch apply** コマンドを実行して、必要なモニターデーモンをデプロイします。

   構文

   ```
   ceph orch apply mon “NUMBER_OF_DAEMONS HOST_NAME_1 HOST_NAME_3”
   ```

   host02 から monitor デーモンを削除する場合は、他のホストにモニターを再デプロイします。

   例

   ```
   [ceph: root@host01 /]# ceph orch apply mon “2 host01 host03”
   ```

検証

- ホスト、デーモン、およびプロセスをリスト表示します。

  構文

  ```
  ceph orch ps --daemon_type=DAEMON_NAME
  ```

  例

  ```
  [ceph: root@host01 /]# ceph orch ps --daemon_type=mon
  ```

関連情報

- 詳細は、**Red Hat Ceph Storage オペレーションガイド**の **コマンドラインインターフェイスを使用した Ceph モニターデーモンのデプロイ** セクションを参照してください。
- 詳細は、**Red Hat Ceph Storage オペレーションガイド** の **サービス仕様を使用した Ceph モニターデーモンのデプロイ** セクションを参照してください。

4.7. 異常なストレージクラスターからの CEPH MONITOR の削除

正常でないストレージクラスターから、**ceph-mon** デーモンを削除できます。正常でないストレージクラスターとは、配置グループが永続的に **active + clean** 状態ではないストレージクラスターのことです。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- Ceph Monitor ノードへの root レベルのアクセス。
- Ceph Monitor ノードが少なくとも 1 台実行している。

手順

1. 存続しているモニターを特定し、ホストにログインします。

   構文

   ```
   ssh root@MONITOR_ID
   ```

   例

   ```
   [root@admin ~]# ssh root@host00
   ```

2. 各 Ceph Monitor ホストにログインし、すべての Ceph Monitor を停止します。

   構文

   ```
   cephadm unit --name DAEMON_NAME.HOSTNAME stop
   ```

   例

   ```
   [root@host00 ~]# cephadm unit --name mon.host00 stop
   ```

3. 拡張デーモンのメンテナンスに適した環境と、デーモンを対話的に実行するための環境を設定します。

   構文

   ```
   cephadm shell --name DAEMON_NAME.HOSTNAME
   ```

   例

   -

```
[root@host00 ~]# cephadm shell --name mon.host00
```

4. **monmap** ファイルのコピーを抽出します。

   構文

   ```
   ceph-mon -i HOSTNAME --extract-monmap TEMP_PATH
   ```

   例

   ```
   [ceph: root@host00 /]# ceph-mon -i host01 --extract-monmap /tmp/monmap

   2022-01-05T11:13:24.440+0000 7f7603bd1700 -1 wrote monmap to /tmp/monmap
   ```

5. Ceph Monitor 以外を削除します。

   構文

   ```
   monmaptool TEMPORARY_PATH --rm HOSTNAME
   ```

   例

   ```
   [ceph: root@host00 /]# monmaptool /tmp/monmap --rm host01
   ```

6. 削除されたモニターを含む存続しているモニターマップを、存続している Ceph モニターに挿入します。

   構文

   ```
   ceph-mon -i HOSTNAME --inject-monmap TEMP_PATH
   ```

   例

   ```
   [ceph: root@host00 /]# ceph-mon -i host00 --inject-monmap /tmp/monmap
   ```

7. 存続しているモニターのみを起動します。

   構文

   ```
   cephadm unit --name DAEMON_NAME.HOSTNAME start
   ```

   例

   ```
   [root@host00 ~]# cephadm unit --name mon.host00 start
   ```

8. モニターがクォーラムを形成していることを確認します。

   例

   ```
   [ceph: root@host00 /]# ceph -s
   ```

9. オプション: 削除された Ceph Monitor のデータディレクトリーを **/var/lib/ceph/CLUSTER_FSID/mon.HOSTNAME** ディレクトリーにアーカイブします。

第5章 CEPH ORCHESTRATOR を使用したマネージャーの管理

ストレージ管理者は、Ceph Orchestrator を使用して追加のマネージャーデーモンをデプロイできます。Cephadm は、ブートストラッププロセス中にブートストラップノードにマネージャーデーモンを自動的にインストールします。

本セクションでは、以下の管理タスクを説明します。

- Ceph Orchestrator を使用したマネージャーデーモンのデプロイ
- Ceph Orchestrator を使用したマネージャーデーモンの削除

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- すべてのノードへの root レベルのアクセス。
- ホストがクラスターに追加されている。

5.1. CEPH ORCHESTRATOR を使用したマネージャーデーモンのデプロイ

Ceph Orchestrator はデフォルトで 2 つの Manager デーモンをデプロイします。コマンドラインインターフェイスで **placement** 仕様を使用して、追加のマネージャーデーモンをデプロイできます。異なる数の Manager デーモンをデプロイするには、別の数を指定します。Manager デーモンがデプロイされるホストを指定しないと、Ceph Orchestrator はホストをランダムに選択し、Manager デーモンをそれらにデプロイします。

注記

デプロイメントごとに少なくとも 3 つの Ceph Manager がデプロイメントに含まれるようにします。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- ホストがクラスターに追加されている。

手順

1. Cephadm シェルにログインします。

   例

   ```
   [root@host01 ~]# cephadm shell
   ```

2. マネージャーデーモンをデプロイする方法は 2 つあります。

方法 1

- 特定のホストセットに配置仕様を使用してマネージャーデーモンをデプロイします。

注記

Red Hat は **--placement** オプションを使用して特定のホストにデプロイすることを推奨します。

構文

```
ceph orch apply mgr --placement=" HOST_NAME_1 HOST_NAME_2 HOST_NAME_3"
```

例

```
[ceph: root@host01 /]# ceph orch apply mgr --placement="host01 host02 host03"
```

方法 2

- ストレージクラスターのホストにマネージャーデーモンを無作為にデプロイします。

  構文

  ```
  ceph orch apply mgr NUMBER_OF_DAEMONS
  ```

  例

  ```
  [ceph: root@host01 /]# ceph orch apply mgr 3
  ```

検証

- サービスをリスト表示します。

  例

  ```
  [ceph: root@host01 /]# ceph orch ls
  ```

- ホスト、デーモン、およびプロセスをリスト表示します。

  構文

  ```
  ceph orch ps --daemon_type=DAEMON_NAME
  ```

  例

  ```
  [ceph: root@host01 /]# ceph orch ps --daemon_type=mgr
  ```

5.2. CEPH ORCHESTRATOR を使用したマネージャーデーモンの削除

ホストからマネージャーデーモンを削除するには、他のホストにデーモンを再デプロイします。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。

- すべてのノードへの root レベルのアクセス。
- ホストがクラスターに追加されている。
- ホストにデプロイされたマネージャーデーモン 1 つ以上。

手順

1. Cephadm シェルにログインします。

   例

   ```
   [root@host01 ~]# cephadm shell
   ```

2. **ceph orch apply** コマンドを実行して、必要なマネージャーデーモンを再デプロイします。

   構文

   ```
   ceph orch apply mgr "NUMBER_OF_DAEMONS HOST_NAME_1 HOST_NAME_3"
   ```

   host02 からマネージャーデーモンを削除する場合は、他のホストにマネージャーデーモンを再デプロイします。

   例

   ```
   [ceph: root@host01 /]# ceph orch apply mgr "2 host01 host03"
   ```

検証

- ホスト、デーモン、およびプロセスをリスト表示します。

  構文

  ```
  ceph orch ps --daemon_type=DAEMON_NAME
  ```

  例

  ```
  [ceph: root@host01 /]# ceph orch ps --daemon_type=mgr
  ```

関連情報

- 詳細は、**Red Hat Ceph Storage オペレーションガイド** の Ceph Orchestrator を使用したマネージャーデーモンのデプロイ セクションを参照してください。

5.3. CEPH MANAGER バランサーモジュールの使用

バランサーは、Ceph Manager (**ceph-mgr**) のモジュールで、OSD 全体の配置グループ (PG) の配置を最適化することで、自動または監視された方法でバランスの取れた分散を実現します。

現在、バランサーモジュールを無効にできません。これをオフにして設定をカスタマイズすることしかできません。

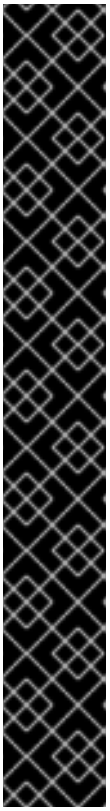
モード

現在、サポートされるバランサーモードが 2 つあります。

- **crush-compat**: CRUSH compat モードは、Ceph Luminous で導入された compat の **weight-set** 機能を使用して、CRUSH 階層のデバイスの別の重みセットを管理します。通常の重みは、デバイスに保存する目的のデータ量を反映するために、デバイスのサイズに設定したままにする必要があります。その後バランサーは、可能な限り目的のディストリビューションに一致するディストリビューションを達成するために、**weight-set** の値を少しずつ増減させ、値を最適化します。PG の配置は擬似ランダムプロセスであるため、配置には自然なばらつきが伴います。重みを最適化することで、バランサーはこの自然なばらつきに対応します。
このモードは、古いクライアントと完全に後方互換性があります。OSDMap および CRUSH マップが古いクライアントと共有されると、バランサーは最適化された weightsff を実際の重みとして提示します。

  このモードの主な制限は、階層のサブツリーが OSD を共有する場合に、バランサーが配置ルールの異なる複数の CRUSH 階層を処理できないことです。この設定では、共有 OSD での領域の使用を管理するのが困難になるため、一般的には推奨されません。そのため、通常、この制限は問題にはなりません。

- **upmap**: Luminous 以降、OSDMap は、通常の CRUSH 配置計算への例外として、個々の OSD の明示的なマッピングを保存できます。これらの **upmap** エントリーにより、PG マッピングを細かく制御できます。この CRUSH モードは、バランスの取れた分散を実現するために、個々の PG の配置を最適化します。多くの場合、この分散は各 OSD の PG 数 +/-1 PG で完璧です。これは割り切れない場合があるためです。



重要

この機能の使用を許可するには、次のコマンドを使用して、luminous 以降のクライアントのみをサポートする必要があることをクラスターに伝える必要があります。

```
[ceph: root@host01 /]# ceph osd set-require-min-compat-client luminous
```

このコマンドは、luminous 以前のクライアントまたはデーモンがモニターに接続されていると失敗します。

既知の問題により、カーネル CephFS クライアントは自身を jewel クライアントとして報告します。この問題を回避するには、**--yes-i-really-mean-it** フラグを使用します。

```
[ceph: root@host01 /]# ceph osd set-require-min-compat-client luminous --
yes-i-really-mean-it
```

使用しているクライアントのバージョンは、以下で確認できます。

```
[ceph: root@host01 /]# ceph features
```

5.3.1. キャパシティーバランサーを使用した Red Hat Ceph クラスターのバランス調整

キャパシティバランサーを使用して、Red Hat Ceph Storage クラスターのバランスをとります。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。

手順

1. バランサーモジュールが有効になっているかどうかを確認します。

   例

   ```
   [ceph: root@host01 /]# ceph mgr module enable balancer
   ```

2. balancer モジュールをオンにします。

   例

   ```
   [ceph: root@host01 /]# ceph balancer on
   ```

3. モードを変更するには、次のコマンドを使用します。デフォルトのモードは **upmap** です。

   例

   ```
   [ceph: root@host01 /]# ceph balancer mode crush-compat
   ```

   または、以下を実行します。

   例

   ```
   [ceph: root@host01 /]# ceph balancer mode upmap
   ```

4. バランサーの現在の状態を確認します。

   例

   ```
   [ceph: root@host01 /]# ceph balancer status
   ```

自動バランシング

デフォルトでは、バランサーモジュールをオンにする場合、自動分散が使用されます。

例

```
[ceph: root@host01 /]# ceph balancer on
```

次のコマンドを使用してバランサーを再度オフにできます。

例

```
[ceph: root@host01 /]# ceph balancer off
```

これには、古いクライアントと後方互換性があり、時間の経過とともにデータディストリビューションに小さな変更を加えて、OSD を同等に利用されるようにする **crush-compat** モードを使用します。

スロットリング

たとえば、OSD が失敗し、システムがまだ修復していない場合などに、クラスターのパフォーマンスが低下する場合は、PG ディストリビューションには調整は行われません。

クラスターが正常な場合、バランサーは変更を調整して、置き間違えた、または移動する必要のある PG の割合がデフォルトで 5% のしきい値を下回るようにします。この割合は、**target_max_misplaced_ratio** の設定を使用して調整できます。たとえば、しきい値を 7% に増やすには、次のコマンドを実行します。

例

```
[ceph: root@host01 /]# ceph config-key set mgr target_max_misplaced_ratio .07
```

自動バランシングの場合:

- 自動バランサーの実行間におけるスリープ時間を秒単位で設定します。

例

```
[ceph: root@host01 /]# ceph config set mgr mgr/balancer/sleep_interval 60
```

- 自動バランシングを開始する時刻を HHMM 形式で設定します。

例

```
[ceph: root@host01 /]# ceph config set mgr mgr/balancer/begin_time 0000
```

- 自動バランシングを終了する時刻を HHMM 形式で設定します。

例

```
[ceph: root@host01 /]# ceph config set mgr mgr/balancer/end_time 2359
```

- 自動バランシングをこの曜日以降に制限します。**0** は日曜日、**1** は月曜日というように、crontab と同じ規則を使用します。

例

```
[ceph: root@host01 /]# ceph config set mgr mgr/balancer/begin_weekday 0
```

- 自動バランシングをこの曜日以前に制限します。**0** は日曜日、**1** は月曜日というように、crontab と同じ規則を使用します。

例

```
[ceph: root@host01 /]# ceph config set mgr mgr/balancer/end_weekday 6
```

- 自動バランシングを制限するプール ID を定義します。これのデフォルトは空の文字列で、すべてのプールが均衡していることを意味します。数値のプール ID は、**ceph osd pool ls detail** コマンドで取得できます。

例

```
[ceph: root@host01 /]# ceph config set mgr mgr/balancer/pool_ids 1,2,3
```

-

監視付き最適化

balancer 操作はいくつかの異なるフェーズに分類されます。

1. **プラン** の構築。
2. 現在の PG ディストリビューションまたは **プラン** 実行後に得られる PG ディストリビューションに対するデータ分散の品質の評価。
3. **プラン** の実行。
   - 現在のディストリビューションを評価し、スコアを付けます。

     例

     ```
     [ceph: root@host01 /]# ceph balancer eval
     ```

   - 単一プールのディストリビューションを評価するには、以下を実行します。

     構文

     ```
     ceph balancer eval POOL_NAME
     ```

     例

     ```
     [ceph: root@host01 /]# ceph balancer eval rbd
     ```

   - 評価の詳細を表示するには、以下を実行します。

     例

     ```
     [ceph: root@host01 /]# ceph balancer eval-verbose ...
     ```

   - 現在設定されているモードを使用してプランを生成するには、以下を実行します。

     構文

     ```
     ceph balancer optimize PLAN_NAME
     ```

     PLAN_NAME は、カスタムプラン名に置き換えます。

     例

     ```
     [ceph: root@host01 /]# ceph balancer optimize rbd_123
     ```

   - プランの内容を表示するには、以下を実行します。

     構文

     ```
     ceph balancer show PLAN_NAME
     ```

     例

```
[ceph: root@host01 /]# ceph balancer show rbd_123
```

- 古いプランを破棄するには、以下を実行します。

  構文

  ```
  ceph balancer rm PLAN_NAME
  ```

  例

  ```
  [ceph: root@host01 /]# ceph balancer rm rbd_123
  ```

- 現在記録されているプランを表示するには、status コマンドを使用します。

  ```
  [ceph: root@host01 /]# ceph balancer status
  ```

- プラン実行後に生じるディストリビューションの品質を計算するには、以下を実行します。

  構文

  ```
  ceph balancer eval PLAN_NAME
  ```

  例

  ```
  [ceph: root@host01 /]# ceph balancer eval rbd_123
  ```

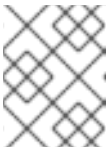
- プランを実行するには、以下を実行します。

  構文

  ```
  ceph balancer execute PLAN_NAME
  ```

  例

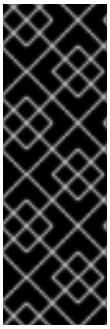
  ```
  [ceph: root@host01 /]# ceph balancer execute rbd_123
  ```



注記

ディストリビューションの改善が想定される場合にのみ、プランを実行します。実行後、計画は破棄されます。

5.3.2. 読み取りバランサーを使用した Red Hat Ceph クラスターのバランス調整 [テクノロジープレビュー]



重要

Read Balancer は、Red Hat Ceph Storage 7.0 専用のテクノロジープレビュー機能です。テクノロジープレビュー機能は、実稼働環境での Red Hat サービスレベルアグリーメント (SLA) ではサポートされておらず、機能的に完全ではない可能性があるため、Red Hat では実稼働環境での使用を推奨していません。テクノロジープレビューの機能は、最新の製品機能をいち早く提供して、開発段階で機能のテストを行いフィードバックを提供していただくことを目的としています。詳細は、Red Hat 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。

アンバランスなプライマリー OSD がある場合は、**osdmaptool** に組み込まれているオフラインオプティマイザーを使用してそれらを更新できます。

Red Hat では、最適な結果を確保するために、読み取りバランサーを実行する前にキャパシティーバランサーを実行することを推奨します。

読み取りバランサーを使用してクラスターのバランスを調整するには、次の手順に従います。

前提条件

- 実行し、容量のバランスが取れた Red Hat Ceph Storage クラスター。
- Red Hat では、最適な結果を確保するために、読み取りバランサーを実行する前に、キャパシティーバランサーを実行して各 OSD の容量のバランスを調整することを推奨します。次の手順を実行して容量のバランスをとります。

  1. osdmap の最新コピーを取得します。

     ```
     [ceph: root@host01 /]# ceph osd  getmap -o map
     ```

  2. アップマップバランサーを実行します。

     ```
     [ceph: root@host01 /]# ospmaptool map –upmap out.txt
     ```

  3. out.txt ファイルには、提案されたソリューションが含まれています。
     この手順のコマンドは、クラスターに変更を適用するために実行する通常の Ceph CLI コマンドです。

     out.txt ファイルに推奨事項がある場合は、次のコマンドを実行します。

     ```
     [ceph: root@host01 /]# source out.txt
     ```

     詳細は、キャパシティーバランサーを使用した IBM Ceph クラスターのバランス調整 を参照してください。

手順

1. 各プールで利用可能な **read_balance_score** を確認します。

   ```
   [ceph: root@host01 /]# ceph osd pool ls detail
   ```

   read_balance_score が 1 を大幅に上回っている場合、プールにはバランスの取れていないプライマリー OSD があります。

同種のクラスターの場合、最適なスコアは Ceil{(PG の数/OSD の数)}/(PG の数/OSD の数]/[PG の数/OSD の数)/(PG の数/OSD の数値)] です。たとえば、32 個の PG と 10 個の OSD を含むプールがある場合は、(PG の数/OSD の数) = 32/10 = 3.2 となります。したがって、すべてのデバイスが同一である場合の最適なスコアは、上限値 3.2 を (PG の数/OSD の数) で割った値、つまり 4/3.2 = 1.25 になります。同じシステム内に 64 PG を持つ別のプールがある場合、最適なスコアは 7/6.4 =1.09375 です。

出力例:

```
$ ceph osd pool ls detail
pool 1 '.mgr' replicated size 3 min_size 1 crush_rule 0 object_hash rjenkins pg_num 1
pgp_num 1 autoscale_mode on last_change 17 flags hashpspool stripe_width 0
pg_num_max 32 pg_num_min 1 application mgr read_balance_score 3.00
pool 2 'cephfs.a.meta' replicated size 3 min_size 1 crush_rule 0 object_hash rjenkins pg_num
16 pgp_num 16 autoscale_mode on last_change 55 lfor 0/0/25 flags hashpspool stripe_width
0 pg_autoscale_bias 4 pg_num_min 16 recovery_priority 5 application cephfs
read_balance_score 1.50
pool 3 'cephfs.a.data' replicated size 3 min_size 1 crush_rule 0 object_hash rjenkins pg_num
128 pgp_num 128 autoscale_mode on last_change 27 lfor 0/0/25 flags hashpspool,bulk
stripe_width 0 application cephfs read_balance_score 1.31
```

2. **osdmap** の最新のコピーを取得します。

```
[ceph: root@host01 /]# ceph osd getmap -o om
```

出力例:

```
got osdmap epoch 56
```

3. オプティマイザーを実行します。
out.txt ファイルには、提案されたソリューションが含まれています。

```
[ceph: root@host01 /]# osdmaptool om --read out.txt --read-pool _POOL_NAME_ [--vstart]
```

出力例:

```
$ osdmaptool om --read out.txt --read-pool cephfs.a.meta
./bin/osdmaptool: osdmap file 'om'
writing upmap command output to: out.txt
---------- BEFORE ------------
 osd.0 | primary affinity: 1 | number of prims: 4
 osd.1 | primary affinity: 1 | number of prims: 8
 osd.2 | primary affinity: 1 | number of prims: 4

read_balance_score of 'cephfs.a.meta': 1.5

---------- AFTER ------------
 osd.0 | primary affinity: 1 | number of prims: 5
 osd.1 | primary affinity: 1 | number of prims: 6
 osd.2 | primary affinity: 1 | number of prims: 5

read_balance_score of 'cephfs.a.meta': 1.13
```

```
num changes: 2
```

4. **out.txt** ファイルには、提案されたソリューションが含まれています。
この手順のコマンドは、クラスターに変更を適用するために実行する通常の Ceph CLI コマンドです。vstart クラスターで作業している場合は、**--vstart** パラメーターを渡すと、CLI コマンドが **./bin/** 接頭辞でフォーマットされるようになります。

```
[ceph: root@host01 /]# source out.txt
```

出力例:

```
$ cat out.txt
ceph osd pg-upmap-primary 2.3 0
ceph osd pg-upmap-primary 2.4 2

$ source out.txt
change primary for pg 2.3 to osd.0
change primary for pg 2.4 to osd.2
```

注記

ceph osd pg-upmap-primary コマンドを初めて実行すると、次のような警告が表示される場合があります。

```
Error EPERM: min_compat_client luminous < reef, which is required for pg-upmap-primary. Try 'ceph osd set-require-min-compat-client reef' before using the new interface
```

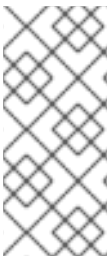
この場合、推奨コマンド **ceph osd set-require-min-compat-client reef** を実行し、クラスターの min-compact-client を調整します。

注記

配置グループ (PG) の数が変更した場合、または OSD がクラスターに追加または削除された場合は、スコアを再チェックしてバランサーを再実行することを検討してください。これらの操作は、プールに対する読み取りバランサーの効果に大きな影響を与える可能性があります。

5.4. CEPHMANAGER アラートモジュールの使用

Ceph Manager アラートモジュールを使用して、Red Hat Ceph Storage クラスターの健全性に関する簡単なアラートメッセージを電子メールで送信できます。



注記

このモジュールは、堅牢な監視ソリューションを目的としたものではありません。Ceph クラスター自体の一部として実行されるため、**ceph-mgr** デーモンに障害が発生するとアラートが送信されないという根本的な制約があります。ただし、このモジュールは、既存の監視インフラストラクチャーが存在しない環境に存在するスタンドアロンクラスターには役立ちます。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- Ceph Monitor ノードへの root レベルのアクセス。

手順

1. Cephadm シェルにログインします。

   例

   ```
   [root@host01 ~]# cephadm shell
   ```

2. アラートモジュールを有効にします。

   例

   ```
   [ceph: root@host01 /]# ceph mgr module enable alerts
   ```

3. アラートモジュールが有効になっていることを確認します。

   例

   ```
   [ceph: root@host01 /]# ceph mgr module ls | more
   {
       "always_on_modules": [
           "balancer",
           "crash",
           "devicehealth",
           "orchestrator",
           "pg_autoscaler",
           "progress",
           "rbd_support",
           "status",
           "telemetry",
           "volumes"
       ],
       "enabled_modules": [
           "alerts",
           "cephadm",
           "dashboard",
           "iostat",
           "nfs",
           "prometheus",
           "restful"
       ]
   ```

4. Simple Mail Transfer Protocol (SMTP) を設定します。

   構文

```
ceph config set mgr mgr/alerts/smtp_host SMTP_SERVER
ceph config set mgr mgr/alerts/smtp_destination RECEIVER_EMAIL_ADDRESS
ceph config set mgr mgr/alerts/smtp_sender SENDER_EMAIL_ADDRESS
```

例

```
[ceph: root@host01 /]# ceph config set mgr mgr/alerts/smtp_host smtp.example.com
[ceph: root@host01 /]# ceph config set mgr mgr/alerts/smtp_destination
example@example.com
[ceph: root@host01 /]# ceph config set mgr mgr/alerts/smtp_sender
example2@example.com
```

5. オプション: デフォルトでは、アラートモジュールは SSL とポート 465 を使用します。

構文

```
ceph config set mgr mgr/alerts/smtp_port PORT_NUMBER
```

例

```
[ceph: root@host01 /]# ceph config set mgr mgr/alerts/smtp_port 587
```

重要

SSL は、Red Hat Ceph Storage 6 クラスターではサポートされていません。アラートの設定中に **smtp_ssl** パラメーターを設定しないでください。

6. SMTP サーバーへの認証:

構文

```
ceph config set mgr mgr/alerts/smtp_user USERNAME
ceph config set mgr mgr/alerts/smtp_password PASSWORD
```

例

```
[ceph: root@host01 /]# ceph config set mgr mgr/alerts/smtp_user admin1234
[ceph: root@host01 /]# ceph config set mgr mgr/alerts/smtp_password admin1234
```

7. オプション: デフォルトでは、SMTP **From** 名は **Ceph** です。これを変更するには、**smtp_from_name** パラメーターを設定します。

構文

```
ceph config set mgr mgr/alerts/smtp_from_name CLUSTER_NAME
```

例

```
[ceph: root@host01 /]# ceph config set mgr mgr/alerts/smtp_from_name 'Ceph Cluster Test'
```

8. オプション: デフォルトでは、アラートモジュールはストレージクラスターの健全性を毎分チェックし、クラスターの健全ステータスに変更があった場合にメッセージを送信します。頻度を変更するには、**interval** パラメーターを設定します。

   構文

   ```
   ceph config set mgr mgr/alerts/interval INTERVAL
   ```

   例

   ```
   [ceph: root@host01 /]# ceph config set mgr mgr/alerts/interval "5m"
   ```

   この例では、間隔は 5 分に設定されています。

9. オプション: アラートをすぐに送信します。

   例

   ```
   [ceph: root@host01 /]# ceph alerts send
   ```

関連情報

- Ceph の健全性メッセージの詳細は、**Red Hat Ceph Storage トラブルシューティングガイド** の **Ceph クラスターの正常性メッセージ** セクションを参照してください。

5.5. CEPH MANAGER クラッシュモジュールの使用

Ceph Manager クラッシュモジュールを使用すると、デーモンの crashdump に関する情報を収集し、これを Red Hat Ceph Storage クラスターに保存して詳細な分析を行うことができます。

デフォルトでは、デーモンの crashdump は **/var/lib/ceph/crash** にダンプされます。オプション **crash dir** で設定できます。クラッシュディレクトリーの名前は、時間、日付、およびランダムに生成される UUID で名前が付けられ、同じ **crash_id** を持つメタデータファイルの **meta** と最近のログファイルが含まれます。

ceph-crash.service を使用して、これらのクラッシュを自動的に送信し、Ceph Monitors で永続化することができます。**ceph-crash.service** は **crashdump** ディレクトリーを監視し、**ceph crash post** でアップロードします。

RECENT_CRASH ヘルスメッセージは、Ceph クラスター内の最も一般的なヘルスメッセージのいずれかとなります。このヘルスメッセージは、1 つ以上の Ceph デーモンが最近クラッシュし、そのクラッシュが管理者によってアーカイブまたは確認されていないことを意味します。これは、ソフトウェアのバグや、障害のあるディスクなどのハードウェアの問題があることを示している可能性があります。オプション **mgr/crash/warn_recent_interval** は、最新の方法の期間 (デフォルトでは 2 週間) を制御します。以下のコマンドを実行して警告を無効にすることができます。

例

```
[ceph: root@host01 /]# ceph config set mgr/crash/warn_recent_interval 0
```

mgr/crash/retain_interval オプションは、自動的にパージされるまでクラッシュレポートを保持する期間を制御します。このオプションのデフォルトは 1 年です。

前提条件

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。

手順

1. crash モジュールが有効になっていることを確認します。

   例

```
[ceph: root@host01 /]# ceph mgr module ls | more
{
    "always_on_modules": [
        "balancer",
        "crash",
        "devicehealth",
        "orchestrator_cli",
        "progress",
        "rbd_support",
        "status",
        "volumes"
    ],
    "enabled_modules": [
        "dashboard",
        "pg_autoscaler",
        "prometheus"
    ]
```

2. クラッシュダンプの保存: **meta** ファイルは、メタとして crash ディレクトリーに保存されている JSON blob です。ceph コマンド **-i -** オプションを呼び出すことができます。これは、stdin から読み取ります。

   例

```
[ceph: root@host01 /]# ceph crash post -i meta
```

3. 新しいクラッシュ情報およびアーカイブされたすべてのクラッシュ情報のタイムスタンプまたは UUID クラッシュ ID を表示します。

   例

```
[ceph: root@host01 /]# ceph crash ls
```

4. すべての新規クラッシュ情報のタイムスタンプまたは UUID クラッシュ ID をリスト表示します。

   例

```
[ceph: root@host01 /]# ceph crash ls-new
```

5. すべての新規クラッシュ情報のタイムスタンプまたは UUID クラッシュ ID をリスト表示します。

   例

```
[ceph: root@host01 /]# ceph crash ls-new
```

6. 保存されたクラッシュ情報の概要を経過時間別にグループ化してリスト表示します。

例

```
[ceph: root@host01 /]# ceph crash stat
8 crashes recorded
8 older than 1 days old:
2022-05-20T08:30:14.533316Z_4ea88673-8db6-4959-a8c6-0eea22d305c2
2022-05-20T08:30:14.590789Z_30a8bb92-2147-4e0f-a58b-a12c2c73d4f5
2022-05-20T08:34:42.278648Z_6a91a778-bce6-4ef3-a3fb-84c4276c8297
2022-05-20T08:34:42.801268Z_e5f25c74-c381-46b1-bee3-63d891f9fc2d
2022-05-20T08:34:42.803141Z_96adfc59-be3a-4a38-9981-e71ad3d55e47
2022-05-20T08:34:42.830416Z_e45ed474-550c-44b3-b9bb-283e3f4cc1fe
2022-05-24T19:58:42.549073Z_b2382865-ea89-4be2-b46f-9a59af7b7a2d
2022-05-24T19:58:44.315282Z_1847afbc-f8a9-45da-94e8-5aef0738954e
```

7. 保存したクラッシュの詳細を表示します。

構文

```
ceph crash info CRASH_ID
```

例

```
[ceph: root@host01 /]# ceph crash info 2022-05-24T19:58:42.549073Z_b2382865-ea89-
4be2-b46f-9a59af7b7a2d
{
    "assert_condition": "session_map.sessions.empty()",
    "assert_file": "/builddir/build/BUILD/ceph-16.1.0-486-g324d7073/src/mon/Monitor.cc",
    "assert_func": "virtual Monitor::~Monitor()",
    "assert_line": 287,
    "assert_msg": "/builddir/build/BUILD/ceph-16.1.0-486-g324d7073/src/mon/Monitor.cc: In
function 'virtual Monitor::~Monitor()' thread 7f67a1aeb700 time 2022-05-
24T19:58:42.545485+0000\n/builddir/build/BUILD/ceph-16.1.0-486-
g324d7073/src/mon/Monitor.cc: 287: FAILED
ceph_assert(session_map.sessions.empty())\n",
    "assert_thread_name": "ceph-mon",
    "backtrace": [
        "/lib64/libpthread.so.0(+0x12b30) [0x7f679678bb30]",
        "gsignal()",
        "abort()",
        "(ceph::__ceph_assert_fail(char const*, char const*, int, char const*)+0x1a9)
[0x7f6798c8d37b]",
        "/usr/lib64/ceph/libceph-common.so.2(+0x276544) [0x7f6798c8d544]",
        "(Monitor::~Monitor()+0xe30) [0x561152ed3c80]",
        "(Monitor::~Monitor()+0xd) [0x561152ed3cdd]",
        "main()",
        "__libc_start_main()",
        "_start()"
    ],
    "ceph_version": "16.2.8-65.el8cp",
    "crash_id": "2022-07-06T19:58:42.549073Z_b2382865-ea89-4be2-b46f-9a59af7b7a2d",
```

```
    "entity_name": "mon.ceph-adm4",
    "os_id": "rhel",
    "os_name": "Red Hat Enterprise Linux",
    "os_version": "8.5 (Ootpa)",
    "os_version_id": "8.5",
    "process_name": "ceph-mon",
    "stack_sig":
"957c21d558d0cba4cee9e8aaf9227b3b1b09738b8a4d2c9f4dc26d9233b0d511",
    "timestamp": "2022-07-06T19:58:42.549073Z",
    "utsname_hostname": "host02",
    "utsname_machine": "x86_64",
    "utsname_release": "4.18.0-240.15.1.el8_3.x86_64",
    "utsname_sysname": "Linux",
    "utsname_version": "#1 SMP Wed Jul 06 03:12:15 EDT 2022"
}
```

8. **KEEP** の日数より古い保存済みクラッシュを削除します。ここで、**KEEP** は整数である必要があります。

   構文

   ```
   ceph crash prune KEEP
   ```

   例

   ```
   [ceph: root@host01 /]# ceph crash prune 60
   ```

9. **RECENT_CRASH** ヘルスチェックで考慮されなくなり、**crash ls-new** の出力に表示されないようにクラッシュレポートをアーカイブします。**crash ls** に表示されます。

   構文

   ```
   ceph crash archive CRASH_ID
   ```

   例

   ```
   [ceph: root@host01 /]# ceph crash archive 2022-05-24T19:58:42.549073Z_b2382865-ea89-
   4be2-b46f-9a59af7b7a2d
   ```

10. すべてのクラッシュレポートをアーカイブします。

    例

    ```
    [ceph: root@host01 /]# ceph crash archive-all
    ```

11. クラッシュダンプを削除します。

    構文

    ```
    ceph crash rm CRASH_ID
    ```

    例

```
[ceph: root@host01 /]# ceph crash rm 2022-05-24T19:58:42.549073Z_b2382865-ea89-4be2-b46f-9a59af7b7a2d
```

関連情報

- Ceph の健全性メッセージの詳細は、**Red Hat Ceph Storage トラブルシューティングガイド** の **Ceph クラスターの正常性メッセージ** セクションを参照してください。

5.6. TELEMETRY モジュール

Telemetry モジュールは、Ceph がどのように使用され、運用中にどのような問題が発生したかを理解するために、ストレージクラスターに関するデータを送信します。データはパブリックダッシュボードで視覚化され、レポートされているクラスターの数、総容量と OSD 数、およびバージョンの分布傾向に関する概要統計が表示されます。

チャネル

Telemetry レポートはいくつかのチャネルに分類され、それぞれに異なる種類の情報が含まれます。Telemetry を有効にすると、個々のチャネルをオンまたはオフにすることができます。

以下に 4 つの異なるチャネルを示します。

- **basic** - デフォルトは **on** です。このチャネルは、次の情報を含む、クラスターに関する基本情報を提供します。
  - クラスターの容量。
  - モニター、マネージャー、OSD、MDS、オブジェクトゲートウェイ、またはその他のデーモンの数。
  - 現在使用されているソフトウェアのバージョン。
  - RADOS プールと Ceph ファイルシステムの数とタイプ。
  - デフォルトから変更した設定オプションの名前 (値は変更されません)。
- **crash** - デフォルトは **on** です。このチャネルは、デーモンのクラッシュに関する情報を提供します。これには次の情報が含まれます。
  - デーモンの種類。
  - デーモンのバージョン。
  - オペレーティングシステム、OS ディストリビューション、およびカーネルのバージョン。
  - Ceph コード内のクラッシュが発生した場所を特定するスタックトレース。
- **device** - デフォルトは **on** です。このチャネルは、匿名化された SMART メトリックを含むデバイスメトリクスに関する情報を提供します。
- **ident** - デフォルトは **off** です。このチャネルは、クラスターの説明や連絡先電子メールアドレスなど、ユーザーが提供したクラスターに関する識別情報を提供します。
- **perf** - デフォルトは **off** です。このチャネルは、クラスターのさまざまなパフォーマンスメトリクスを提供し、次の目的で使用できます。
  - クラスター全体の健全性を明らかにします。

- ワークロードのパターンを特定します。
- レイテンシー、スロットル、メモリー管理、その他の同様の問題に関するトラブルシューティングを行います。
- デーモンごとにクラスターのパフォーマンスを監視します。

報告されるデータには、プール名、オブジェクト名、オブジェクトの内容、ホスト名、デバイスのシリアル番号などの機密データは含まれません。

これには、クラスターのデプロイ方法、Ceph バージョン、ホスト分散、およびプロジェクトが Ceph の使用方法をより深く理解するのに役立つその他のパラメーターに関するカウンターと統計が含まれています。

データは安全に保護され、https://telemetry.ceph.com に送信されます。

テレメトリーを有効にする

チャネルを有効にする前に、テレメトリーが **on** になっていることを確認してください。

- テレメトリーを有効にします。

```
ceph telemetry on
```

チャネルの有効化と無効化

- 個々のチャネルを有効または無効にします。

```
ceph telemetry enable channel basic
ceph telemetry enable channel crash
ceph telemetry enable channel device
ceph telemetry enable channel ident
ceph telemetry enable channel perf

ceph telemetry disable channel basic
ceph telemetry disable channel crash
ceph telemetry disable channel device
ceph telemetry disable channel ident
ceph telemetry disable channel perf
```

- 複数のチャネルを有効または無効にします。

```
ceph telemetry enable channel basic crash device ident perf
ceph telemetry disable channel basic crash device ident perf
```

- すべてのチャネルをまとめて有効または無効にします。

```
ceph telemetry enable channel all
ceph telemetry disable channel all
```

サンプルレポート

- レポートされたデータをいつでも確認するには、サンプルレポートを生成します。

```
ceph telemetry show
```

- Telemetry が **off** の場合は、サンプルレポートをプレビューします。

```
ceph telemetry preview
```

数百以上の OSD を持つストレージクラスターのサンプルレポートを生成するには、より長い時間がかかります。

- プライバシーを保護するために、デバイスレポートは個別に生成され、ホスト名やデバイスのシリアル番号などのデータは匿名化されます。デバイスの Telemetry は別のエンドポイントに送信され、デバイスデータを特定のクラスターに関連付けません。デバイスレポートを表示するには、次のコマンドを実行します。

```
ceph telemetry show-device
```

- Telemetry が **off** の場合は、サンプルデバイスレポートをプレビューします。

```
ceph telemetry preview-device
```

- Telemetry を **on** にして、両方のレポートについて 1 つの出力を取得します。

```
ceph telemetry show-all
```

- Telemetry を **off** にして、両方のレポートについて 1 つの出力を取得します。

```
ceph telemetry preview-all
```

- チャネルごとにサンプルレポートを生成します。

構文

```
ceph telemetry show CHANNEL_NAME
```

- チャネルごとにサンプルレポートのプレビューを生成します。

構文

```
ceph telemetry preview CHANNEL_NAME
```

コレクション

コレクションは、チャネル内で収集されるデータのさまざまな側面です。

- コレクションをリストします。

```
ceph telemetry collection ls
```

- 登録しているコレクションと、新しく利用可能なコレクションの違いを確認してください。

```
ceph telemetry diff
```

- 最新のコレクションに登録します。

構文

```
ceph telemetry on
ceph telemetry enable channel CHANNEL_NAME
```

Interval

このモジュールは、デフォルトで 24 時間ごとに新しいレポートをコンパイルして送信します。

- 間隔を調整します。

  構文

  ```
  ceph config set mgr mgr/telemetry/interval INTERVAL
  ```

  例

  ```
  [ceph: root@host01 /]# ceph config set mgr mgr/telemetry/interval 72
  ```

  この例では、レポートは 3 日 (72 時間) ごとに生成されます。

ステータス

- 現在の設定を表示します。

  ```
  ceph telemetry status
  ```

Telemetry を手動で送信する

- Telemetry データをアドホックベースで送信します。

  ```
  ceph telemetry send
  ```

  Telemetry が無効になっている場合は、**--license sharing-1-0** を **ceph telemetry send** コマンドに追加します。

プロキシー経由で Telemetry を送信する

- クラスターが設定済みの Telemetry エンドポイントに直接接続できない場合は、HTTP/HTTPS プロキシーサーバーを設定できます。

  構文

  ```
  ceph config set mgr mgr/telemetry/proxy PROXY_URL
  ```

  例

  ```
  [ceph: root@host01 /]# ceph config set mgr mgr/telemetry/proxy https://10.0.0.1:8080
  ```

  コマンドにユーザーパスを含めることができます。

  例

```
[ceph: root@host01 /]# ceph config set mgr mgr/telemetry/proxy https://10.0.0.1:8080
```

連絡先と説明

- オプション: 連絡先と説明をレポートに追加します。

  構文

```
ceph config set mgr mgr/telemetry/contact '_CONTACT_NAME_'
ceph config set mgr mgr/telemetry/description '_DESCRIPTION_'
ceph config set mgr mgr/telemetry/channel_ident true
```

  例

```
[ceph: root@host01 /]# ceph config set mgr mgr/telemetry/contact 'John Doe
<john.doe@example.com>'
[ceph: root@host01 /]# ceph config set mgr mgr/telemetry/description 'My first Ceph cluster'
[ceph: root@host01 /]# ceph config set mgr mgr/telemetry/channel_ident true
```

  ident フラグが有効になっている場合、その詳細はリーダーボードに表示されません。

リーダーボード

- 公開ダッシュボードのリーダーボードに参加します。

  例

```
[ceph: root@host01 /]# ceph config set mgr mgr/telemetry/leaderboard true
```

  リーダーボードには、ストレージクラスターに関する基本情報が表示されます。このボードには、総ストレージ容量と OSD の数が含まれます。

Telemetry を無効にする

- Telemetry はいつでも無効にできます。

  例

```
ceph telemetry off
```

第6章 CEPH ORCHESTRATOR を使用した OSD の管理

ストレージ管理者は、Ceph Orchestrators を使用して Red Hat Ceph Storage クラスターの OSD を管理できます。

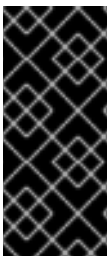
6.1. CEPH OSD

Red Hat Ceph Storage クラスターが稼働している場合は、ランタイム時に OSD をストレージクラスターに追加できます。

Ceph OSD は、通常 1 つのストレージドライブおよびノード内の関連付けられたジャーナル用に 1 つの **ceph-osd** デーモンで設定されます。ノードに複数のストレージドライブがある場合は、ドライブごとに 1 つの **ceph-osd** デーモンをマッピングします。

Red Hat は、クラスターの容量を定期的に確認して、ストレージ容量の最後に到達するかどうかを確認することを推奨します。ストレージクラスターが **ほぼ完全** の比率に達すると、1 つ以上の OSD を追加してストレージクラスターの容量を拡張します。

Red Hat Ceph Storage クラスターのサイズを縮小したり、ハードウェアを置き換える場合は、ランタイム時に OSD を削除することも可能です。ノードに複数のストレージドライブがある場合には、そのドライブ用に **ceph-osd** デーモンのいずれかを削除する必要もあります。通常、ストレージクラスターの容量を確認して、容量の上限に達したかどうかを確認することが推奨されます。ストレージクラスターが **ほぼ完全** の比率ではないことを OSD を削除する場合。



重要

OSD を追加する前に、ストレージクラスターが **完全な** 比率を超えないようにします。ストレージクラスターが **ほぼ完全な** 比率に達した後に OSD の障害が発生すると、ストレージクラスターが **完全な** 比率を超過する可能性があります。Ceph は、ストレージ容量の問題を解決するまでデータを保護するための書き込みアクセスをブロックします。**完全な** 比率の影響を考慮せずに OSD を削除しないでください。

6.2. CEPH OSD ノードの設定

OSD を使用するプールのストレージストラテジーとして同様に Ceph OSD とサポートするハードウェアを設定します。Ceph は、一貫性のあるパフォーマンスプロファイルを確保するために、プール全体でハードウェアを統一します。最適なパフォーマンスを得るには、同じタイプまたはサイズのドライブのある CRUSH 階層を検討してください。

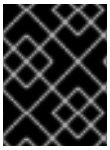
異なるサイズのドライブを追加する場合は、それに応じて重量を調整してください。OSD を CRUSH マップに追加する場合は、新規 OSD の重みを考慮してください。ハードドライブの容量は、1 年あたり約 40% 増加するため、新しい OSD ノードはストレージクラスターの古いノードよりも大きなハードドライブを持つ可能性があります。つまり、重みが大きくなる可能性があります。

新たにインストールを行う前に、インストールガイド の**Red Hat Ceph Storage のインストール要件**の章を確認します。

6.3. OSD メモリーの自動チューニング

OSD デーモンは、**osd_memory_target** 設定オプションに基づいてメモリー消費を調整します。**osd_memory_target** オプションは、システムで利用可能な RAM に基づいて OSD メモリーを設定します。

Red Hat Ceph Storage が他のサービスとメモリーを共有しない専用ノードにデプロイされている場合、**cephadm** は RAM の合計量とデプロイされた OSD の数に基づいて OSD ごとの消費を自動的に調整します。



重要

デフォルトでは、Red Hat Ceph Storage 6.0 で **osd_memory_target_autotune** パラメーターは **true** に設定されます。

構文

```
ceph config set osd osd_memory_target_autotune true
```

OSD の追加や OSD の置き換えなど、クラスターのメンテナンスのためにストレージクラスターを Red Hat Ceph Storage 6.0 にアップグレードした後、Red Hat は **osd_memory_target_autotune** パラメーターを **true** に設定し、システムメモリーごとに osd メモリーを自動調整することを推奨します。

Cephadm は、**mgr/cephadm/autotune_memory_target_ratio** の割合で始まります。これはデフォルトでは、システムの合計 RAM 容量の **0.7** になります。これから、非 OSDS や **osd_memory_target_autotune** が false の OSD などの自動調整されないデーモンによって消費されるメモリー分を引き、残りの OSD で割ります。

osd_memory_target パラメーターは、以下のように計算されます。

構文

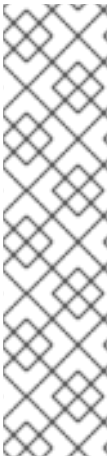
```
osd_memory_target = TOTAL_RAM_OF_THE_OSD * (1048576) * (autotune_memory_target_ratio) / NUMBER_OF_OSDS_IN_THE_OSD_NODE - (SPACE_ALLOCATED_FOR_OTHER_DAEMONS)
```

SPACE_ALLOCATED_FOR_OTHER_DAEMONS には、任意で以下のデーモン領域の割り当てを含めることができます。

- Alertmanager: 1 GB
- Grafana: 1 GB
- Ceph Manager: 4 GB
- Ceph Monitor: 2 GB
- Node-exporter: 1 GB
- Prometheus: 1 GB

たとえば、ノードに OSD が 24 個あり、251 GB の RAM 容量がある場合、 **osd_memory_target** は **7860684936** になります。

最後のターゲットは、オプションとともに設定データベースに反映されます。**MEM LIMIT** 列の **ceph orch ps** の出力で、制限と各デーモンによって消費される現在のメモリーを確認できます。



注記

Red Hat Ceph Storage 6.0 では、**osd_memory_target_autotune** のデフォルト設定 **true** は、コンピュートサービスと Ceph ストレージサービスが共存するハイパーコンバージドインフラストラクチャーでは適切ではありません。ハイパーコンバージドインフラストラクチャーでは、**autotune_memory_target_ratio** を **0.2** に設定して、Ceph のメモリー消費を減らすことができます。

例

```
[ceph: root@host01 /]# ceph config set mgr
mgr/cephadm/autotune_memory_target_ratio 0.2
```

ストレージクラスターで OSD の特定のメモリーターゲットを手動で設定できます。

例

```
[ceph: root@host01 /]# ceph config set osd.123 osd_memory_target 7860684936
```

ストレージクラスターで OSD ホストの特定のメモリーターゲットを手動で設定できます。

構文

```
ceph config set osd/host:HOSTNAME osd_memory_target TARGET_BYTES
```

例

```
[ceph: root@host01 /]# ceph config set osd/host:host01 osd_memory_target 1000000000
```

注記

osd_memory_target_autotune を有効にすると、既存の手動の OSD メモリーターゲット設定が上書きされます。**osd_memory_target_autotune** オプションまたはその他の同様のオプションが有効になっている場合でもデーモンメモリーがチューニングされないようにするには、ホストに **_no_autotune_memory** ラベルを設定します。

構文

```
ceph orch host label add HOSTNAME _no_autotune_memory
```

自動チューニングオプションを無効にし、特定のメモリーターゲットを設定して、OSD をメモリー自動チューニングから除外できます。

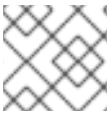
例

```
[ceph: root@host01 /]# ceph config set osd.123 osd_memory_target_autotune false
[ceph: root@host01 /]# ceph config set osd.123 osd_memory_target 16G
```

6.4. CEPH OSD デプロイメント用のデバイスのリスト表示

Ceph Orchestrator を使用して OSD をデプロイする前に、利用可能なデバイスのリストを確認することができます。コマンドは、Cephadm によって検出可能なデバイスのリストを出力するために使用されます。以下の条件がすべて満たされると、ストレージデバイスが利用可能であるとみなされます。

- デバイスにはパーティションがない。
- デバイスは LVM 状態でない。
- デバイスをマウントしてはいけない。
- デバイスにはファイルシステムを含めることはできない。
- デバイスには Ceph BlueStore OSD を含めることはできない。
- デバイスは 5 GB を超える必要がある。



注記

Ceph は、利用できないデバイスに OSD をプロビジョニングしません。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- ホストがクラスターに追加されている。
- すべてのマネージャーおよびモニターデーモンがデプロイされている。

手順

1. Cephadm シェルにログインします。

   例

   ```
   [root@host01 ~]# cephadm shell
   ```

2. OSD をデプロイするために利用可能なデバイスをリスト表示します。

   構文

   ```
   ceph orch device ls [--hostname=HOSTNAME_1 HOSTNAME_2] [--wide] [--refresh]
   ```

   例

   ```
   [ceph: root@host01 /]# ceph orch device ls --wide --refresh
   ```

   --wide オプションを使用すると、デバイスが OSD として使用できない可能性がある理由など、デバイスに関連するすべての詳細が提供されます。このオプションは、NVMe デバイスをサポートしません。

3. 任意手順: **ceph orch device ls** の出力で **Health**、**Ident** および **Fault** フィールドを有効にするには、以下のコマンドを実行します。

注記

これらのフィールドは **libstoragemgmt** ライブラリーによってサポートされ、現時点では SCSI、SAS、SATA デバイスをサポートします。

a. Cephadm シェルの外部で root ユーザーとして、ハードウェアと **libstoragemgmt** ライブラリーとの互換性を確認して、サービスの予期しない中断を回避します。

例

```
[root@host01 ~]# cephadm shell lsmcli ldl
```

この出力で、それぞれの **SCSI VPD 0x83** ID で **Health Status** が **Good** と表示されます。

注記

この情報を取得しないと、フィールドを有効にした場合にデバイスで異常な挙動が発生する可能性があります。

b. Cephadm シェルに再度ログインし、**libstoragemgmt** サポートを有効にします。

例

```
[root@host01 ~]# cephadm shell
[ceph: root@host01 /]# ceph config set mgr mgr/cephadm/device_enhanced_scan true
```

これが有効化されると、**ceph orch device ls** を実行した場合、**Health** フィールドの出力は **Good** になります。

検証

- デバイスをリスト表示します。

例

```
[ceph: root@host01 /]# ceph orch device ls
```

6.5. CEPH OSD デプロイメントのデバイスの消去

OSD をデプロイする前に、利用可能なデバイスのリストを確認する必要があります。デバイスに空き容量がない場合は、そのデバイスを消去してデバイス上のデータを消去します。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- ホストがクラスターに追加されている。
- すべてのマネージャーおよびモニターデーモンがデプロイされている。

手順

1. Cephadm シェルにログインします。

   例

   ```
   [root@host01 ~]# cephadm shell
   ```

2. OSD をデプロイするために利用可能なデバイスをリスト表示します。

   構文

   ```
   ceph orch device ls [--hostname=HOSTNAME_1 HOSTNAME_2] [--wide] [--refresh]
   ```

   例

   ```
   [ceph: root@host01 /]# ceph orch device ls --wide --refresh
   ```

3. デバイスのデータをクリアします。

   構文

   ```
   ceph orch device zap HOSTNAME FILE_PATH --force
   ```

   例

   ```
   [ceph: root@host01 /]# ceph orch device zap host02 /dev/sdb --force
   ```

検証

- デバイスに容量があることを確認します。

  例

  ```
  [ceph: root@host01 /]# ceph orch device ls
  ```

  Available の下のフィールドが **Yes** であることを確認できます。

関連情報

- 詳細は、**Red Hat Ceph Storage オペレーションガイド** の Ceph OSD デプロイメント用のデバイスのリスト表示 セクションを参照してください。

6.6. すべての利用可能なデバイスへの CEPH OSD のデプロイ

すべての OSDS を利用可能なすべてのデバイスにデプロイできます。Cephadm により、Ceph Orchestrator は利用可能な未使用のストレージデバイスで OSD を検出およびデプロイできます。

使用可能なすべてのデバイスに OSD をデプロイメントするには、**unmanaged** パラメーターを指定せずにコマンドを実行し、その後 OSD が作成されないようにパラメーターを指定してコマンドを再実行します。

注記

--all-available-devices を使用した OSD のデプロイメントは、通常、小規模なクラスターに使用されます。大規模なクラスターの場合は、OSD 仕様ファイルを使用します。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- ホストがクラスターに追加されている。
- すべてのマネージャーおよびモニターデーモンがデプロイされている。

手順

1. Cephadm シェルにログインします。

   例

   ```
   [root@host01 ~]# cephadm shell
   ```

2. OSD をデプロイするために利用可能なデバイスをリスト表示します。

   構文

   ```
   ceph orch device ls [--hostname=HOSTNAME_1 HOSTNAME_2] [--wide] [--refresh]
   ```

   例

   ```
   [ceph: root@host01 /]# ceph orch device ls --wide --refresh
   ```

3. すべての利用可能なデバイスに OSD をデプロイします。

   例

   ```
   [ceph: root@host01 /]# ceph orch apply osd --all-available-devices
   ```

   ceph orch apply の影響は永続的であり、Orchestrator は自動的にデバイスを見つけ、それをクラスターに追加し、新しい OSD を作成します。これは、以下の条件下で実行されます。

   - 新しいディスクまたはドライブがシステムに追加される。
   - 既存のディスクまたはドライブは消去される。
   - OSD は削除され、デバイスは消去される。
     --unmanaged パラメーターを使用して、利用可能なすべてのデバイスで OSD の自動作成を無効にできます。

     例

     ```
     [ceph: root@host01 /]# ceph orch apply osd --all-available-devices --unmanaged=true
     ```

パラメーター **--unmanaged** を **true** に設定すると、OSD の作成が無効になり、新しい OSD サービスを適用しても変更はありません。

注記

コマンド **ceph orch daemon add** は、新しい OSD を作成しますが、OSD サービスを追加しません。

検証

- サービスをリスト表示します。

  例

  ```
  [ceph: root@host01 /]# ceph orch ls
  ```

- ノードとデバイスの詳細を表示します。

  例

  ```
  [ceph: root@host01 /]# ceph osd tree
  ```

関連情報

- **Red Hat Ceph Storage オペレーションガイド** の **Ceph OSD デプロイメント用のデバイスのリスト表示** セクションを参照してください。

6.7. 特定のデバイスおよびホストへの CEPH OSD のデプロイ

Ceph Orchestrator を使用して、特定のデバイスおよびホストにすべての Ceph OSD をデプロイできます。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- ホストがクラスターに追加されている。
- すべてのマネージャーおよびモニターデーモンがデプロイされている。

手順

1. Cephadm シェルにログインします。

   例

   ```
   [root@host01 ~]# cephadm shell
   ```

2. OSD をデプロイするために利用可能なデバイスをリスト表示します。

   構文

   ```
   ceph orch device ls [--hostname=HOSTNAME_1 HOSTNAME_2] [--wide] [--refresh]
   ```

例

```
[ceph: root@host01 /]# ceph orch device ls --wide --refresh
```

3. 特定のデバイスおよびホストに OSD をデプロイします。

構文

```
ceph orch daemon add osd HOSTNAME:DEVICE_PATH
```

例

```
[ceph: root@host01 /]# ceph orch daemon add osd host02:/dev/sdb
```

LVM レイヤーを使用せずに raw 物理デバイスに ODS をデプロイするには、**--method raw** オプションを使用します。

構文

```
ceph orch daemon add osd --method raw HOSTNAME:DEVICE_PATH
```

例

```
[ceph: root@host01 /]# ceph orch daemon add osd --method raw host02:/dev/sdb
```

注記

別の DB または WAL デバイスがある場合、ブロックと DB デバイスまたは WAL デバイスの比率は **1:1 でなければなりません**。

検証

- サービスをリスト表示します。

例

```
[ceph: root@host01 /]# ceph orch ls osd
```

- ノードとデバイスの詳細を表示します。

例

```
[ceph: root@host01 /]# ceph osd tree
```

- ホスト、デーモン、およびプロセスをリスト表示します。

構文

```
ceph orch ps --service_name=SERVICE_NAME
```

例

```
[ceph: root@host01 /]# ceph orch ps --service_name=osd
```

関連情報

- **Red Hat Ceph Storage オペレーションガイド** の Ceph OSD デプロイメント用のデバイスのリスト表示 セクションを参照してください。

6.8. OSD をデプロイするための高度なサービス仕様およびフィルター

タイプ OSD のサービス仕様は、ディスクのプロパティーを使用してクラスターレイアウトを記述する 1 つの方法です。これは、デバイスの名前やパスの指定を把握せずに、必要な設定でどのディスクを OSD にするか Ceph に知らせる抽象的な方法をユーザーに提供します。各デバイスおよび各ホストに対して、**yaml** ファイルまたは **json** ファイルを定義します。

OSD 仕様の一般設定

- **service_type**: 'osd': これは、OSDS の作成には必須です。
- **service_id**: 希望するサービス名または ID を使用します。OSD のセットは、仕様ファイルを使用して作成されます。この名前は、すべての OSD を管理し、Orchestrator サービスを表すために使用されます。
- **placement**: これは、OSD をデプロイする必要のあるホストを定義するために使用されます。以下のオプションで使用できます。
  - **host_pattern**: '*' - ホストの選択に使用するホスト名パターン。
  - **label**: 'osd_host' - OSD をデプロイメントする必要があるホストで使用されるラベル。
  - **hosts**: 'host01', 'host02': OSD をデプロイする必要があるホスト名の明示的なリスト。
- **selection of devices**: OSD が作成されるデバイス。これにより、OSD を異なるデバイスから分離することができます。3 つのコンポーネントを持つ BlueStore OSD のみを作成できます。
  - OSD データ: すべての OSD データが含まれます。
  - WAL: BlueStore 内部ジャーナルまたはログ先行書き込み
  - DB: BlueStore 内部メタデータ
- **data_devices**: OSD をデプロイするデバイスを定義します。この場合、OSD は同じ場所に配置されるスキーマに作成されます。フィルターを使用して、デバイスおよびフォルダーを選択できます。
- **wal_devices**: WAL OSD に使用されるデバイスを定義します。フィルターを使用して、デバイスおよびフォルダーを選択できます。
- **db_devices**: DB OSD のデバイスを定義します。フィルターを使用して、デバイスおよびフォルダーを選択できます。
- **encrypted**: **True** または **False** のいずれかに設定される OSD の情報を暗号化するオプションのパラメーター

- **unmanaged**: オプションのパラメーターで、デフォルトでは False に設定されます。Orchestrator で OSD サービスを管理しない場合は、これを True に設定します。
- **block_wal_size**: ユーザー定義の値 (バイト単位) 。
- **block_db_size**: ユーザー定義の値 (バイト単位) 。
- **osds_per_device**: デバイスごとに複数の OSD をデプロイするためのユーザー定義値。
- **method**: OSD が LVM レイヤーで作成されるかどうかを指定するオプションのパラメーター。LVM レイヤーを含まない raw 物理デバイス上に OSD を作成する場合は **raw** に設定します。別の DB または WAL デバイスがある場合、ブロックと DB デバイスまたは WAL デバイスの比率は **1:1 でなければなりません**。

デバイスを指定するためのフィルター

フィルターは、**data_devices**、**wal_devices**、および **db_devices** パラメーターとともに使用されます。

フィルターの名前	説明	構文	例
Model	ターゲット固有のディスク。**lsblk -o NAME,FSTYPE,LABEL,MOUNTPOINT,SIZE,MODEL** コマンドまたは **smartctl -i /DEVIVE_PATH** を実行すると、モデルの詳細を取得できます。	Model: **DISK_MODEL_NAME**	Model: MC-55-44-XZ
Vendor	ターゲット固有のディスク	Vendor: **DISK_VENDOR_NAME**	Vendor: Vendor Cs
Size Specification	正確なサイズのディスク	size: **EXACT**	size: '10G'
Size Specification	範囲内にあるディスクサイズ	size: **LOW:HIGH**	size: '10G:40G'
Size Specification	サイズがこれ以下のディスク	size: **:HIGH**	size: ':10G'
Size Specification	サイズがこれ以上のディスク	size: **LOW:**	size: '40G:'

Rotational	ディスクの回転属性。1 はローテーションするすべてのディスクと一致し、0 はローテーションされないすべてのディスクと一致します。rotational =0 の場合、OSD は SSD または NVME で設定されます。rotational=1 の場合、OSD は HDD で設定されます。	rotational: 0 or 1	rotational: 0
All	利用可能なディスクをすべて考慮します。	all: true	all: true
Limiter	有効なフィルターが指定されていても、一致するディスクの量を制限する場合は、'limit' ディレクティブを使用できます。これは、最後の手段としてのみ使用してください。	limit: **NUMBER**	limit: 2

注記

同じホストに同じ場所にないコンポーネントを持つ OSD を作成するには、使用される異なるタイプのデバイスを指定し、デバイスが同じホスト上になければなりません。

注記

OSD のデプロイに使用されるデバイスは、**libstoragemgmt** によってサポートされる必要があります。

関連情報

- **Red Hat Ceph Storage オペレーションガイド** の **詳細仕様を使用した Ceph CSD のデプロイメント** セクションを参照してください。
- **libstoragemgmt** の詳細は、**Red Hat Ceph Storage オペレーションガイド** の **Ceph OSD デプロイメント用のデバイスのリスト表示** セクションを参照してください。

6.9. 高度なサービス仕様を使用した CEPH OSD のデプロイ

タイプ OSD のサービス仕様は、ディスクのプロパティーを使用してクラスターレイアウトを記述する 1 つの方法です。これは、デバイスの名前やパスの指定を把握せずに、必要な設定でどのディスクを OSD にするか Ceph に知らせる抽象的な方法をユーザーに提供します。

各デバイスおよび各ホストに OSD をデプロイするには、**yaml** ファイルまたは **json** ファイルを定義します。

前提条件

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- ホストがクラスターに追加されている。
- すべてのマネージャーおよびモニターデーモンがデプロイされている。

手順

1. モニターノードで、**osd_spec.yaml** ファイルを作成します。

   例

   ```
   [root@host01 ~]# touch osd_spec.yaml
   ```

2. **osd_spec.yml** ファイルを編集し、以下の詳細を含めます。

   構文

   ```
   service_type: osd
   service_id: SERVICE_ID
   placement:
     host_pattern: '*' # optional
   data_devices: # optional
     model: DISK_MODEL_NAME # optional
     paths:
     - /DEVICE_PATH
   osds_per_device: NUMBER_OF_DEVICES # optional
   db_devices: # optional
     size: # optional
     all: true # optional
     paths:
      - /DEVICE_PATH
   encrypted: true
   ```

   a. 単純なシナリオ: これらのケースでは、すべてのノードが同じ設定になっています。

      例

      ```
      service_type: osd
      service_id: osd_spec_default
      placement:
        host_pattern: '*'
      data_devices:
        all: true
        paths:
        - /dev/sdb
      encrypted: true
      ```

      例

      ```
      service_type: osd
      service_id: osd_spec_default
      placement:
      ```

```
  host_pattern: '*'
data_devices:
  size: '80G'
db_devices:
  size: '40G:'
  paths:
   - /dev/sdc
```

b. 単純なシナリオ: この場合、すべてのノードが、LVM レイヤーを使用せず、raw モードで作成された OSD デバイスと同じ設定になっています。

例

```
service_type: osd
service_id: all-available-devices
encrypted: "true"
method: raw
placement:
  host_pattern: "*"
data_devices:
  all: "true"
```

c. 高度なシナリオ: これにより、すべての HDD を専用の DB または WAL デバイスとして割り当てられた 2 つの SSD のある **data_devices** として使用し、目的のレイアウトを作成します。残りの SSD は、NVME ベンダーが専用の DB または WAL デバイスとして割り当てられている **data_devices** です。

例

```
service_type: osd
service_id: osd_spec_hdd
placement:
  host_pattern: '*'
data_devices:
  rotational: 0
db_devices:
  model: Model-name
  limit: 2
---
service_type: osd
service_id: osd_spec_ssd
placement:
  host_pattern: '*'
data_devices:
  model: Model-name
db_devices:
  vendor: Vendor-name
```

d. 不均一のノードを使用する高度なシナリオ: これは、host_pattern キーに応じて、異なる OSD 仕様を異なるホストに適用します。

例

```
service_type: osd
```

```
service_id: osd_spec_node_one_to_five
placement:
  host_pattern: 'node[1-5]'
data_devices:
  rotational: 1
db_devices:
  rotational: 0
---
service_type: osd
service_id: osd_spec_six_to_ten
placement:
  host_pattern: 'node[6-10]'
data_devices:
  model: Model-name
db_devices:
  model: Model-name
```

e. 専用の WAL および DB デバイスを使用した高度なシナリオ:

例

```
service_type: osd
service_id: osd_using_paths
placement:
  hosts:
    - host01
    - host02
data_devices:
  paths:
    - /dev/sdb
db_devices:
  paths:
    - /dev/sdc
wal_devices:
  paths:
    - /dev/sdd
```

f. デバイスごとに複数の OSD がある高度なシナリオ:

例

```
service_type: osd
service_id: multiple_osds
placement:
  hosts:
    - host01
    - host02
osds_per_device: 4
data_devices:
  paths:
    - /dev/sdb
```

g. 事前に作成されたボリュームの場合、**osd_spec.yaml** ファイルを編集して、次の詳細を含めます。

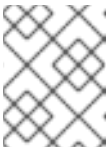
構文

```
service_type: osd
service_id: SERVICE_ID
placement:
  hosts:
    - HOSTNAME
data_devices: # optional
  model: DISK_MODEL_NAME # optional
  paths:
  - /DEVICE_PATH
db_devices: # optional
  size: # optional
  all: true # optional
  paths:
   - /DEVICE_PATH
```

例

```
service_type: osd
service_id: osd_spec
placement:
  hosts:
    - machine1
data_devices:
  paths:
    - /dev/vg_hdd/lv_hdd
db_devices:
  paths:
    - /dev/vg_nvme/lv_nvme
```

h. ID による OSD の場合は、**osd_spec.yaml** ファイルを編集して次の詳細を含めます。



注記

この設定は、Red Hat Ceph Storage 5.3z1 以降のリリースに適用されます。以前のリリースの場合は、事前に作成された LVM を使用してください。

構文

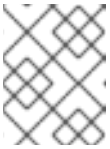
```
service_type: osd
service_id: OSD_BY_ID_HOSTNAME
placement:
  hosts:
    - HOSTNAME
data_devices: # optional
  model: DISK_MODEL_NAME # optional
  paths:
  - /DEVICE_PATH
db_devices: # optional
  size: # optional
  all: true # optional
  paths:
   - /DEVICE_PATH
```

-

例

```
service_type: osd
service_id: osd_by_id_host01
placement:
  hosts:
    - host01
data_devices:
  paths:
    - /dev/disk/by-id/scsi-0QEMU_QEMU_HARDDISK_drive-scsi0-0-0-5
db_devices:
  paths:
    - /dev/disk/by-id/nvme-nvme.1b36-31323334-51454d55204e564d65204374726c-
00000001
```

i. パス別の OSD の場合は、**osd_spec.yaml** ファイルを編集して次の詳細を含めます。



注記

この設定は、Red Hat Ceph Storage 5.3z1 以降のリリースに適用されます。以前のリリースの場合は、事前に作成された LVM を使用してください。

構文

```
service_type: osd
service_id: OSD_BY_PATH_HOSTNAME
placement:
  hosts:
    - HOSTNAME
data_devices: # optional
  model: DISK_MODEL_NAME # optional
  paths:
  - /DEVICE_PATH
db_devices: # optional
  size: # optional
  all: true # optional
  paths:
   - /DEVICE_PATH
```

例

```
service_type: osd
service_id: osd_by_path_host01
placement:
  hosts:
    - host01
data_devices:
  paths:
    - /dev/disk/by-path/pci-0000:0d:00.0-scsi-0:0:0:4
db_devices:
  paths:
    - /dev/disk/by-path/pci-0000:00:02.0-nvme-1
```

3. YAML ファイルをコンテナー内のディレクトリーにマウントします。

   例

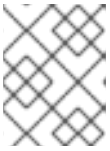
   ```
   [root@host01 ~]# cephadm shell --mount osd_spec.yaml:/var/lib/ceph/osd/osd_spec.yaml
   ```

4. そのディレクトリーに移動します。

   例

   ```
   [ceph: root@host01 /]# cd /var/lib/ceph/osd/
   ```

5. OSD をデプロイする前にドライランを実行します。

   注記

   この手順は、デーモンをデプロイしなくても、デプロイメントのプレビューを提供します。

   例

   ```
   [ceph: root@host01 osd]# ceph orch apply -i osd_spec.yaml --dry-run
   ```

6. サービス仕様を使用して OSD をデプロイします。

   構文

   ```
   ceph orch apply -i FILE_NAME.yml
   ```

   例

   ```
   [ceph: root@host01 osd]# ceph orch apply -i osd_spec.yaml
   ```

検証

- サービスをリスト表示します。

  例

  ```
  [ceph: root@host01 /]# ceph orch ls osd
  ```

- ノードとデバイスの詳細を表示します。

  例

  ```
  [ceph: root@host01 /]# ceph osd tree
  ```

関連情報

- **Red Hat Ceph Storage オペレーションガイド** の **OSD をデプロイするための高度なサービス仕様およびフィルター** セクションを参照してください。

6.10. CEPH ORCHESTRATOR を使用した OSD デーモンの削除

Cephadm を使用して、OSD をクラスターから削除できます。

クラスターから OSD を削除するには、2 つの手順を実行します。

1. クラスターからすべての配置グループ (PG) を退避します。
2. PG のない OSD をクラスターから削除します。

--zap オプションにより、ボリュームグループ、論理ボリューム、LVM メタデータが削除されました。

注記

OSD を削除した後、OSD がデプロイメントされたドライブが再び使用可能になった場合、既存のドライブグループの仕様と一致する場合、**cephadm`** はこれらのドライブにさらに OSD をデプロイメントしようとする可能性があります。削除する OSD を仕様とともにデプロイメントし、削除後に新しい OSD をドライブにデプロイメントしたくない場合は、削除する前にドライブグループの仕様を変更します。OSD のデプロイメント中に **--all-available-devices** オプションを使用した場合は、**unmanaged: true** を設定して、新しいドライブをまったく取得しないようにします。他のデプロイメントの場合は、仕様を変更してください。詳細は、**高度なサービス仕様を使用した Ceph OSD のデプロイ** を参照してください。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- ホストがクラスターに追加されている。
- Ceph Monitor、Ceph Manager、および Ceph OSD デーモンがストレージクラスターにデプロイされます。

手順

1. Cephadm シェルにログインします。

   例

   ```
   [root@host01 ~]# cephadm shell
   ```

2. OSD を削除する必要があるデバイスとノードを確認します。

   例

   ```
   [ceph: root@host01 /]# ceph osd tree
   ```

3. OSD を削除します。

   構文

   ```
   ceph orch osd rm OSD_ID [--replace] [--force] --zap
   ```

   例

```
[ceph: root@host01 /]# ceph orch osd rm 0 --zap
```

注記

--replace などのオプションを指定せずにストレージクラスターから OSD を削除すると、デバイスはストレージクラスターから完全に削除されます。OSD のデプロイに同じデバイスを使用する必要がある場合は、ストレージクラスターにデバイスを追加する前に、最初にデバイスをザッピングする必要があります。

4. オプション: 特定のノードから複数の OSD を削除するには、次のコマンドを実行します。

   構文

   ```
   ceph orch osd rm OSD_ID OSD_ID --zap
   ```

   例

   ```
   [ceph: root@host01 /]# ceph orch osd rm 2 5 --zap
   ```

5. OSD の削除のステータスを確認します。

   例

   ```
   [ceph: root@host01 /]# ceph orch osd rm status
   OSD  HOST   STATE                    PGS  REPLACE  FORCE  ZAP   DRAIN STARTED AT
   9    host01 done, waiting for purge    0  False    False  True  2023-06-06 17:50:50.525690
   10   host03 done, waiting for purge    0  False    False  True  2023-06-06 17:49:38.731533
   11   host02 done, waiting for purge    0  False    False  True  2023-06-06 17:48:36.641105
   ```

   OSD に PG が残っていない場合は廃止され、クラスターから削除されます。

検証

- Ceph OSD が削除されたデバイスおよびノードの詳細を確認します。

  例

  ```
  [ceph: root@host01 /]# ceph osd tree
  ```

関連情報

- 詳細は、**Red Hat Ceph Storage オペレーションガイド** の **すべての利用可能なデバイスへの Ceph OSD のデプロイ** セクションを参照してください。
- 詳細は、**Red Hat Ceph Storage オペレーションガイド** の **特定のデバイスおよびホストへの Ceph OSD のデプロイ** セクションを参照してください。
- デバイスのスペースをクリアする方法の詳細は、**Red Hat Ceph Storage オペレーションガイド** の **Ceph OSD デプロイメントのデバイスの消去** セクションを参照してください。

6.11. CEPH ORCHESTRATOR を使用した OSD の置き換え

ディスクに障害が発生した場合は、物理ストレージデバイスを交換し、同じ OSD ID を再利用することで、CRUSH マップを再設定する必要がなくなります。

--replace オプションを使用して、クラスターから OSD を置き換えることができます。

注記

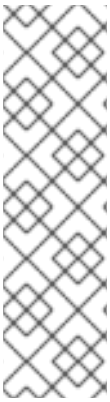
単一の OSD を置き換える場合は、**特定のデバイスおよびホストへの Ceph OSD のデプロイ** を参照してください。利用可能なすべてのデバイスに OSD をデプロイする場合は、**すべての利用可能なデバイスへの Ceph OSD のデプロイ** を参照してください。

このオプションは、**ceph orch rm** コマンドを使用して OSD ID を保存します。OSD は CRUSH 階層から永続的に削除されませんが、**destroyed** フラグが割り当てられます。このフラグは、次の OSD デプロイメントで再利用できる OSD ID を判断するために使用されます。**destroyed** フラグは、次の OSD デプロイメントで再利用される OSD ID を判断するために使用されます。

rm コマンドと同様に、クラスターから OSD を置き換えるには、次の 2 つの手順が必要です。

- クラスターからすべての配置グループ (PG) を退避します。
- クラスターから PG のない OSD を削除します。

デプロイメントに OSD 仕様を使用する場合、新たに追加されたディスクには置き換えられたディスクの OSD ID が割り当てられます。



注記

OSD を削除した後、OSD がデプロイメントされたドライブが再び使用可能になった場合、既存のドライブグループの仕様と一致する場合、**cephadm** はこれらのドライブにさらに OSD をデプロイメントしようとする可能性があります。削除する OSD を仕様とともにデプロイメントし、削除後に新しい OSD をドライブにデプロイメントしたくない場合は、削除する前にドライブグループの仕様を変更します。OSD のデプロイメント中に **--all-available-devices** オプションを使用した場合は、**unmanaged: true** を設定して、新しいドライブをまったく取得しないようにします。他のデプロイメントの場合は、仕様を変更してください。詳細は、**高度なサービス仕様を使用した Ceph OSD のデプロイ** を参照してください。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- ホストがクラスターに追加されている。
- Monitor、Manager、および OSD デーモンはストレージクラスターにデプロイされます。
- 削除された OSD を置き換える新しい OSD は、OSD が削除されたのと同じホストで作成する必要があります。

手順

1. Cephadm シェルにログインします。

   例

   ```
   [root@host01 ~]# cephadm shell
   ```

2. 今後の参照のために、OSD 設定のマッピングを必ずダンプして保存してください。

例

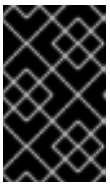
```
[ceph: root@node /]# ceph osd metadata -f plain | grep device_paths
"device_paths": "sde=/dev/disk/by-path/pci-0000:03:00.0-scsi-0:0:0:1,sdi=/dev/disk/by-path/pci-0000:03:00.0-scsi-0:1:0:1",
"device_paths": "sde=/dev/disk/by-path/pci-0000:03:00.0-scsi-0:0:0:1,sdf=/dev/disk/by-path/pci-0000:03:00.0-scsi-0:1:0:1",
"device_paths": "sdd=/dev/disk/by-path/pci-0000:03:00.0-scsi-0:0:0:2,sdg=/dev/disk/by-path/pci-0000:03:00.0-scsi-0:1:0:2",
"device_paths": "sdd=/dev/disk/by-path/pci-0000:03:00.0-scsi-0:0:0:2,sdh=/dev/disk/by-path/pci-0000:03:00.0-scsi-0:1:0:2",
"device_paths": "sdd=/dev/disk/by-path/pci-0000:03:00.0-scsi-0:0:0:2,sdk=/dev/disk/by-path/pci-0000:03:00.0-scsi-0:1:0:2",
"device_paths": "sdc=/dev/disk/by-path/pci-0000:03:00.0-scsi-0:0:0:3,sdl=/dev/disk/by-path/pci-0000:03:00.0-scsi-0:1:0:3",
"device_paths": "sdc=/dev/disk/by-path/pci-0000:03:00.0-scsi-0:0:0:3,sdj=/dev/disk/by-path/pci-0000:03:00.0-scsi-0:1:0:3",
"device_paths": "sdc=/dev/disk/by-path/pci-0000:03:00.0-scsi-0:0:0:3,sdm=/dev/disk/by-path/pci-0000:03:00.0-scsi-0:1:0:3",
[.. output omitted ..]
```

3. OSD を置き換える必要のあるデバイスとノードを確認します。

例

```
[ceph: root@host01 /]# ceph osd tree
```

4. OSD を置き換えます。



重要

ストレージクラスターに **health_warn** またはそれに関連するその他のエラーがある場合は、データの損失を回避するために、OSD を交換する前にエラーを確認して修正してください。

構文

```
ceph orch osd rm OSD_ID --replace [--force]
```

--force オプションは、ストレージクラスターで進行中の操作がある場合に使用できます。

例

```
[ceph: root@host01 /]# ceph orch osd rm 0 --replace
```

5. OSD 置き換えのステータスを確認します。

例

```
[ceph: root@host01 /]# ceph orch osd rm status
```

6. オーケストレーターを停止して、既存の OSD 仕様を適用します。

   例

   ```
   [ceph: root@node /]# ceph orch pause
   [ceph: root@node /]# ceph orch status
   Backend: cephadm
   Available: Yes
   Paused: Yes
   ```

7. 削除された OSD デバイスをザップします。

   例

   ```
   [ceph: root@node /]# ceph orch device zap node.example.com /dev/sdi --force
   zap successful for /dev/sdi on node.example.com

   [ceph: root@node /]# ceph orch device zap node.example.com /dev/sdf --force
   zap successful for /dev/sdf on node.example.com
   ```

8. Orcestrator を一時停止モードから再開する

   例

   ```
   [ceph: root@node /]# ceph orch resume
   ```

9. OSD 置き換えのステータスを確認します。

   例

   ```
   [ceph: root@node /]# ceph osd tree
   ID  CLASS  WEIGHT   TYPE NAME     STATUS  REWEIGHT  PRI-AFF
   -1         0.77112  root default
   -3         0.77112      host node
    0    hdd  0.09639          osd.0     up   1.00000  1.00000
    1    hdd  0.09639          osd.1     up   1.00000  1.00000
    2    hdd  0.09639          osd.2     up   1.00000  1.00000
    3    hdd  0.09639          osd.3     up   1.00000  1.00000
    4    hdd  0.09639          osd.4     up   1.00000  1.00000
    5    hdd  0.09639          osd.5     up   1.00000  1.00000
    6    hdd  0.09639          osd.6     up   1.00000  1.00000
    7    hdd  0.09639          osd.7     up   1.00000  1.00000
    [.. output omitted ..]
   ```

検証

- Ceph OSD が置き換えられるデバイスとノードの詳細を確認します。

  例

  ```
  [ceph: root@host01 /]# ceph osd tree
  ```

  置き換えたものと同じ ID を持つ OSD が同じホスト上で実行されていることがわかります。

- 新しくデプロイされた OSD の **db_device** が置き換えられた **db_device** であることを確認します。

  例

  ```
  [ceph: root@host01 /]# ceph osd metadata 0 | grep bluefs_db_devices
  "bluefs_db_devices": "nvme0n1",

  [ceph: root@host01 /]# ceph osd metadata 1 | grep bluefs_db_devices
  "bluefs_db_devices": "nvme0n1",
  ```

関連情報

- 詳細は、**Red Hat Ceph Storage オペレーションガイド** の **すべての利用可能なデバイスへの Ceph OSD のデプロイ** セクションを参照してください。
- 詳細は、**Red Hat Ceph Storage オペレーションガイド** の **特定のデバイスおよびホストへの Ceph OSD のデプロイ** セクションを参照してください。

6.12. OSD を事前に作成された LVM に置き換える

ceph-volume lvm zap コマンドを使用して OSD をパージした後、ディレクトリーが存在しない場合は、事前に作成された LVM を使用して OSD を OSd サービス仕様ファイルに置き換えることができます。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- 失敗した OSD

手順

1. Cephadm シェルにログインします。

   例

   ```
   [root@host01 ~]# cephadm shell
   ```

2. OSD を削除します。

   構文

   ```
   ceph orch osd rm OSD_ID [--replace]
   ```

   例

   ```
   [ceph: root@host01 /]# ceph orch osd rm 8 --replace
   Scheduled OSD(s) for removal
   ```

3. OSD が破壊されていることを確認します。

   例

```
[ceph: root@host01 /]# ceph osd tree

ID   CLASS  WEIGHT   TYPE NAME       STATUS     REWEIGHT  PRI-AFF
 -1         0.32297  root default
 -9         0.05177      host host10
  3    hdd  0.01520          osd.3          up   1.00000  1.00000
 13    hdd  0.02489          osd.13         up   1.00000  1.00000
 17    hdd  0.01169          osd.17         up   1.00000  1.00000
-13         0.05177      host host11
  2    hdd  0.01520          osd.2          up   1.00000  1.00000
 15    hdd  0.02489          osd.15         up   1.00000  1.00000
 19    hdd  0.01169          osd.19         up   1.00000  1.00000
 -7         0.05835      host host12
 20    hdd  0.01459          osd.20         up   1.00000  1.00000
 21    hdd  0.01459          osd.21         up   1.00000  1.00000
 22    hdd  0.01459          osd.22         up   1.00000  1.00000
 23    hdd  0.01459          osd.23         up   1.00000  1.00000
 -5         0.03827      host host04
  1    hdd  0.01169          osd.1          up   1.00000  1.00000
  6    hdd  0.01129          osd.6          up   1.00000  1.00000
  7    hdd  0.00749          osd.7          up   1.00000  1.00000
  9    hdd  0.00780          osd.9          up   1.00000  1.00000
 -3         0.03816      host host05
  0    hdd  0.01169          osd.0          up   1.00000  1.00000
  8    hdd  0.01129          osd.8    destroyed         0  1.00000
 12    hdd  0.00749          osd.12         up   1.00000  1.00000
 16    hdd  0.00769          osd.16         up   1.00000  1.00000
-15         0.04237      host host06
  5    hdd  0.01239          osd.5          up   1.00000  1.00000
 10    hdd  0.01540          osd.10         up   1.00000  1.00000
 11    hdd  0.01459          osd.11         up   1.00000  1.00000
-11         0.04227      host host07
  4    hdd  0.01239          osd.4          up   1.00000  1.00000
 14    hdd  0.01529          osd.14         up   1.00000  1.00000
 18    hdd  0.01459          osd.18         up   1.00000  1.00000
```

4. **ceph-volume** コマンドを使用して、OSD をザップして削除します。

構文

```
ceph-volume lvm zap --osd-id OSD_ID
```

例

```
[ceph: root@host01 /]# ceph-volume lvm zap --osd-id 8

Zapping: /dev/vg1/data-lv2
Closing encrypted path /dev/mapper/l4D6ql-Prji-IzH4-dfhF-xzuf-5ETl-jNRcXC
Running command: /usr/sbin/cryptsetup remove /dev/mapper/l4D6ql-Prji-IzH4-dfhF-xzuf-
5ETl-jNRcXC
Running command: /usr/bin/dd if=/dev/zero of=/dev/vg1/data-lv2 bs=1M count=10
conv=fsync
 stderr: 10+0 records in
```

```
10+0 records out
 stderr: 10485760 bytes (10 MB, 10 MiB) copied, 0.034742 s, 302 MB/s
Zapping successful for OSD: 8
```

5. OSD トポロジーを確認します。

   例

   ```
   [ceph: root@host01 /]# ceph-volume lvm list
   ```

6. 特定の OSD トポロジーに対応する仕様ファイルを使用して OSD を再作成します。

   例

   ```
   [ceph: root@host01 /]# cat osd.yml
   service_type: osd
   service_id: osd_service
   placement:
     hosts:
     - host03
   data_devices:
     paths:
     - /dev/vg1/data-lv2
   db_devices:
     paths:
      - /dev/vg1/db-lv1
   ```

7. 更新された仕様ファイルを適用します。

   例

   ```
   [ceph: root@host01 /]# ceph orch apply -i osd.yml
   Scheduled osd.osd_service update...
   ```

8. OSD が戻っていることを確認します。

   例

   ```
   [ceph: root@host01 /]# ceph -s
   [ceph: root@host01 /]# ceph osd tree
   ```

6.13. 非共存シナリオでの OSD の交換

非コロケーションシナリオで OSD に障害が発生した場合は、WAL/DB デバイスを交換できます。DB デバイスと WAL デバイスの手順は同じです。DB デバイスの場合は **db_devices** の下の **paths** を編集し、WAL デバイスの場合は **wal_devices** の下の **paths** を編集する必要があります。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- デーモンは同じ場所に配置されません。

- 失敗した OSD

手順

1. クラスター内のデバイスを特定します。

   例

```
[root@host01 ~]# lsblk

NAME                                                                  MAJ:MIN RM  SIZE RO
TYPE MOUNTPOINT
sda                                                                     8:0    0   20G  0 disk
├─sda1                                                                  8:1    0    1G  0 part
/boot
└─sda2                                                                  8:2    0   19G  0 part
  ├─rhel-root                                                         253:0    0   17G  0 lvm  /
  └─rhel-swap                                                         253:1    0    2G  0 lvm
[SWAP]
sdb                                                                     8:16   0   10G  0 disk
└─ceph--5726d3e9--4fdb--4eda--b56a--3e0df88d663f-osd--block--3ceb89ec--87ef--46b4--
99c6--2a56bac09ff0 253:2    0   10G  0 lvm
sdc                                                                     8:32   0   10G  0 disk
└─ceph--d7c9ab50--f5c0--4be0--a8fd--e0313115f65c-osd--block--37c370df--1263--487f--
a476--08e28bdbcd3c 253:4    0   10G  0 lvm
sdd                                                                     8:48   0   10G  0 disk
├─ceph--1774f992--44f9--4e78--be7b--b403057cf5c3-osd--db--31b20150--4cbc--4c2c--
9c8f--6f624f3bfd89    253:7    0  2.5G  0 lvm
└─ceph--1774f992--44f9--4e78--be7b--b403057cf5c3-osd--db--1bee5101--dbab--4155--
a02c--e5a747d38a56    253:9    0  2.5G  0 lvm
sde                                                                     8:64   0   10G  0 disk
sdf                                                                     8:80   0   10G  0 disk
└─ceph--412ee99b--4303--4199--930a--0d976e1599a2-osd--block--3a99af02--7c73--4236--
9879--1fad1fe6203d 253:6    0   10G  0 lvm
sdg                                                                     8:96   0   10G  0 disk
└─ceph--316ca066--aeb6--46e1--8c57--f12f279467b4-osd--block--58475365--51e7--42f2--
9681--e0c921947ae6 253:8    0   10G  0 lvm
sdh                                                                     8:112  0   10G  0 disk
├─ceph--d7064874--66cb--4a77--a7c2--8aa0b0125c3c-osd--db--0dfe6eca--ba58--438a--
9510--d96e6814d853    253:3    0    5G  0 lvm
└─ceph--d7064874--66cb--4a77--a7c2--8aa0b0125c3c-osd--db--26b70c30--8817--45de--
8843--4c0932ad2429    253:5    0    5G  0 lvm
sr0
```

2. Cephadm シェルにログインします。

   例

```
[root@host01 ~]# cephadm shell
```

3. OSD とその DB デバイスを特定します。

   例

```
[ceph: root@host01 /]# ceph-volume lvm list /dev/sdh


====== osd.2 =======

  [db]          /dev/ceph-d7064874-66cb-4a77-a7c2-8aa0b0125c3c/osd-db-0dfe6eca-ba58-
438a-9510-d96e6814d853

      block device              /dev/ceph-5726d3e9-4fdb-4eda-b56a-3e0df88d663f/osd-block-
3ceb89ec-87ef-46b4-99c6-2a56bac09ff0
      block uuid                GkWLoo-f0jd-Apj2-Zmwj-ce0h-OY6J-UuW8aD
      cephx lockbox secret
      cluster fsid              fa0bd9dc-e4c4-11ed-8db4-001a4a00046e
      cluster name              ceph
      crush device class
      db device                 /dev/ceph-d7064874-66cb-4a77-a7c2-8aa0b0125c3c/osd-db-
0dfe6eca-ba58-438a-9510-d96e6814d853
      db uuid                   6gSPoc-L39h-afN3-rDl6-kozT-AX9S-XR20xM
      encrypted                 0
      osd fsid                  3ceb89ec-87ef-46b4-99c6-2a56bac09ff0
      osd id                    2
      osdspec affinity          non-colocated
      type                      db
      vdo                       0
      devices                   /dev/sdh

====== osd.5 =======

  [db]          /dev/ceph-d7064874-66cb-4a77-a7c2-8aa0b0125c3c/osd-db-26b70c30-8817-
45de-8843-4c0932ad2429

      block device              /dev/ceph-d7c9ab50-f5c0-4be0-a8fd-e0313115f65c/osd-block-
37c370df-1263-487f-a476-08e28bdbcd3c
      block uuid                Eay3I7-fcz5-AWvp-kRcI-mJaH-n03V-Zr0wmJ
      cephx lockbox secret
      cluster fsid              fa0bd9dc-e4c4-11ed-8db4-001a4a00046e
      cluster name              ceph
      crush device class
      db device                 /dev/ceph-d7064874-66cb-4a77-a7c2-8aa0b0125c3c/osd-db-
26b70c30-8817-45de-8843-4c0932ad2429
      db uuid                   mwSohP-u72r-DHcT-BPka-piwA-lSwx-w24N0M
      encrypted                 0
      osd fsid                  37c370df-1263-487f-a476-08e28bdbcd3c
      osd id                    5
      osdspec affinity          non-colocated
      type                      db
      vdo                       0
      devices                   /dev/sdh
```

4. **osds.yaml** ファイルで、**unmaged** パラメーターを **true** に設定します。それ以外の場合、**cephadm** は OSD を再デプロイします。

   例

```
[ceph: root@host01 /]# cat osds.yml
```

```
service_type: osd
service_id: non-colocated
unmanaged: true
placement:
  host_pattern: 'ceph*'
data_devices:
  paths:
   - /dev/sdb
   - /dev/sdc
   - /dev/sdf
   - /dev/sdg
db_devices:
  paths:
   - /dev/sdd
   - /dev/sdh
```

5. 更新された仕様ファイルを適用します。

   例

```
[ceph: root@host01 /]# ceph orch apply -i osds.yml

Scheduled osd.non-colocated update...
```

6. ステータスを確認します。

   例

```
[ceph: root@host01 /]# ceph orch ls

NAME           PORTS        RUNNING  REFRESHED  AGE  PLACEMENT
alertmanager   ?:9093,9094      1/1  9m ago     4d   count:1
crash                           3/4  4d ago     4d   *
grafana        ?:3000           1/1  9m ago     4d   count:1
mgr                             1/2  4d ago     4d   count:2
mon                             3/5  4d ago     4d   count:5
node-exporter  ?:9100           3/4  4d ago     4d   *
osd.non-colocated               8  4d ago     5s   <unmanaged>
prometheus     ?:9095           1/1  9m ago     4d   count:1
```

7. OSD を削除します。バックエンドサービスを削除するには **--zap** オプションを使用し、OSD ID を保持するには **--replace** オプションを使用してください。

   例

```
[ceph: root@host01 /]# ceph orch osd rm 2 5 --zap --replace
Scheduled OSD(s) for removal
```

8. ステータスを確認します。

   例

```
[ceph: root@host01 /]# ceph osd df tree | egrep -i "ID|host02|osd.2|osd.5"
```

```
ID  CLASS  WEIGHT   REWEIGHT  SIZE     RAW USE  DATA     OMAP  META     AVAIL
%USE   VAR   PGS  STATUS     TYPE NAME
-5         0.04877         -   55 GiB   15 GiB  4.1 MiB   0 B   60 MiB  40 GiB  27.27  1.17    -
host02
 2    hdd  0.01219   1.00000   15 GiB  5.0 GiB  996 KiB   0 B   15 MiB  10 GiB  33.33  1.43    0
destroyed          osd.2
 5    hdd  0.01219   1.00000   15 GiB  5.0 GiB  1.0 MiB   0 B   15 MiB  10 GiB  33.33  1.43    0
destroyed          osd.5
```

9. **osds.yaml** 仕様ファイルを編集して **アンマネージド** パラメーターを **false** に変更し、デバイスが物理的に交換された後にパスが変更された場合は DB デバイスへのパスを置き換えます。

   例

```
[ceph: root@host01 /]# cat osds.yml
service_type: osd
service_id: non-colocated
unmanaged: false
placement:
  host_pattern: 'ceph01*'
data_devices:
  paths:
   - /dev/sdb
   - /dev/sdc
   - /dev/sdf
   - /dev/sdg
db_devices:
  paths:
   - /dev/sdd
   - /dev/sde
```

   上記の例では、**/dev/sdh** は **/dev/sde** に置き換えられます。

   重要

   同じホスト仕様ファイルを使用して単一の OSD ノード上の障害のある DB デバイスを交換する場合は、OSD ノードのみを指定するように **host_pattern** オプションを変更します。そうしなければ、デプロイメントが失敗し、他のホストで新しい DB デバイスが見つかりません。

10. **--dry-run** オプションを指定して仕様ファイルを再適用し、OSD が新しい DB デバイスでデプロイされることを確認します。

    例

```
[ceph: root@host01 /]# ceph orch apply -i osds.yml --dry-run
WARNING! Dry-Runs are snapshots of a certain point in time and are bound
to the current inventory setup. If any of these conditions change, the
preview will be invalid. Please make sure to have a minimal
timeframe between planning and applying the specs.
####################
SERVICESPEC PREVIEWS
####################
+---------+------+--------+-------------+
```

```
|SERVICE  |NAME  |ADD_TO  |REMOVE_FROM  |
+---------+------+--------+-------------+
+---------+------+--------+-------------+
################
OSDSPEC PREVIEWS
################
+---------+-------+-------+----------+----------+-----+
|SERVICE  |NAME   |HOST   |DATA      |DB        |WAL  |
+---------+-------+-------+----------+----------+-----+
|osd      |non-colocated  |host02  |/dev/sdb  |/dev/sde  |-    |
|osd      |non-colocated  |host02  |/dev/sdc  |/dev/sde  |-    |
+---------+-------+-------+----------+----------+-----+
```

11. 仕様ファイルを適用します。

    例

    ```
    [ceph: root@host01 /]# ceph orch apply -i osds.yml
    Scheduled osd.non-colocated update...
    ```

12. OSD が再デプロイメントされていることを確認します。

    例

    ```
    [ceph: root@host01 /]# ceph osd df tree | egrep -i "ID|host02|osd.2|osd.5"

    ID  CLASS  WEIGHT   REWEIGHT  SIZE    RAW USE  DATA    OMAP  META    AVAIL
    %USE   VAR   PGS  STATUS  TYPE NAME
    -5         0.04877         -   55 GiB   15 GiB  4.5 MiB   0 B   60 MiB  40 GiB  27.27  1.17    -
    host host02
     2    hdd  0.01219   1.00000   15 GiB  5.0 GiB  1.1 MiB   0 B   15 MiB  10 GiB  33.33  1.43    0
    up          osd.2
     5    hdd  0.01219   1.00000   15 GiB  5.0 GiB  1.1 MiB   0 B   15 MiB  10 GiB  33.33  1.43    0
    up          osd.5
    ```

検証

1. OSDS が再デプロイされている OSD ホストから、OSDS が新しい DB デバイス上にあるかどうかを確認します。

   例

   ```
   [ceph: root@host01 /]# ceph-volume lvm list /dev/sde

   ====== osd.2 =======

     [db]          /dev/ceph-15ce813a-8a4c-46d9-ad99-7e0845baf15e/osd-db-1998a02e-5e67-
   42a9-b057-e02c22bbf461

         block device              /dev/ceph-a4afcb78-c804-4daf-b78f-3c7ad1ed0379/osd-block-
   564b3d2f-0f85-4289-899a-9f98a2641979
         block uuid                ITPVPa-CCQ5-BbFa-FZCn-FeYt-c5N4-ssdU41
         cephx lockbox secret
         cluster fsid              fa0bd9dc-e4c4-11ed-8db4-001a4a00046e
   ```

```
    cluster name              ceph
    crush device class
    db device                 /dev/ceph-15ce813a-8a4c-46d9-ad99-7e0845baf15e/osd-db-
1998a02e-5e67-42a9-b057-e02c22bbf461
    db uuid                   HF1bYb-fTK7-0dcB-CHzW-xvNn-dCym-KKdU5e
    encrypted                 0
    osd fsid                  564b3d2f-0f85-4289-899a-9f98a2641979
    osd id                    2
    osdspec affinity          non-colocated
    type                      db
    vdo                       0
    devices                   /dev/sde

====== osd.5 =======

  [db]          /dev/ceph-15ce813a-8a4c-46d9-ad99-7e0845baf15e/osd-db-6c154191-846d-
4e63-8c57-fc4b99e182bd

    block device              /dev/ceph-b37c8310-77f9-4163-964b-f17b4c29c537/osd-block-
b42a4f1f-8e19-4416-a874-6ff5d305d97f
    block uuid                0LuPoz-ao7S-UL2t-BDIs-C9pl-ct8J-xh5ep4
    cephx lockbox secret
    cluster fsid              fa0bd9dc-e4c4-11ed-8db4-001a4a00046e
    cluster name              ceph
    crush device class
    db device                 /dev/ceph-15ce813a-8a4c-46d9-ad99-7e0845baf15e/osd-db-
6c154191-846d-4e63-8c57-fc4b99e182bd
    db uuid                   SvmXms-iWkj-MTG7-VnJj-r5Mo-Moiw-MsbqVD
    encrypted                 0
    osd fsid                  b42a4f1f-8e19-4416-a874-6ff5d305d97f
    osd id                    5
    osdspec affinity          non-colocated
    type                      db
    vdo                       0
    devices                   /dev/sde
```

6.14. CEPH ORCHESTRATOR を使用した OSD の削除停止

削除用にキューに置かれた OSD のみの削除を停止することができます。これにより、OSD の初期状態がリセットされ、削除キューから削除されます。

OSD の削除処理中である場合は、プロセスを停止することはできません。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- ホストがクラスターに追加されている。
- Monitor、Manager デーモン、および OSD デーモンがクラスターにデプロイされている。
- 開始した OSD プロセスを削除する。

手順

1. Cephadm シェルにログインします。

   例

   ```
   [root@host01 ~]# cephadm shell
   ```

2. 削除するために開始された OSD のデバイスとノードを確認します。

   例

   ```
   [ceph: root@host01 /]# ceph osd tree
   ```

3. キューに置かれた OSD の削除を停止します。

   構文

   ```
   ceph orch osd rm stop OSD_ID
   ```

   例

   ```
   [ceph: root@host01 /]# ceph orch osd rm stop 0
   ```

4. OSD の削除のステータスを確認します。

   例

   ```
   [ceph: root@host01 /]# ceph orch osd rm status
   ```

検証

- Ceph OSD を削除するためにキューに入れられたデバイスおよびノードの詳細を確認します。

  例

  ```
  [ceph: root@host01 /]# ceph osd tree
  ```

関連情報

- 詳細は、**Red Hat Ceph Storage オペレーションガイド** の Ceph Orchestrator を使用した OSD デーモンの削除 セクションを参照してください。

6.15. CEPH ORCHESTRATOR を使用した OSD のアクティブ化

ホストのオペレーティングシステムが再インストールされた場合に、クラスターで OSD をアクティブにすることができます。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- ホストがクラスターに追加されている。

- Monitor、Manager、および OSD デーモンがストレージクラスターにデプロイされている。

手順

1. Cephadm シェルにログインします。

   例

   ```
   [root@host01 ~]# cephadm shell
   ```

2. ホストのオペレーティングシステムを再インストールしたら、OSD をアクティベートします。

   構文

   ```
   ceph cephadm osd activate HOSTNAME
   ```

   例

   ```
   [ceph: root@host01 /]# ceph cephadm osd activate host03
   ```

検証

- サービスをリスト表示します。

  例

  ```
  [ceph: root@host01 /]# ceph orch ls
  ```

- ホスト、デーモン、およびプロセスをリスト表示します。

  構文

  ```
  ceph orch ps --service_name=SERVICE_NAME
  ```

  例

  ```
  [ceph: root@host01 /]# ceph orch ps --service_name=osd
  ```

6.16. データ移行の監視

OSD を CRUSH マップに追加または削除すると、Ceph は配置グループを新規または既存の OSD に移行してデータのリバランスを開始します。**ceph-w** コマンドを使用して、データの移行を確認できます。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- 最近 OSD を追加または削除した。

手順

1. データの移行を確認するには、以下を実行します。

   例

   ```
   [ceph: root@host01 /]# ceph -w
   ```

2. 配置グループのステータスが **active+clean** から **active, some degraded objects** し、最後に移行の完了時に **active+clean** に変わるのを確認します。

3. ユーティリティーを終了するには、**Ctrl + C** を押します。

6.17. 配置グループの再計算

配置グループ (PG) は、利用可能な OSD にまたがるプールデータの分散を定義します。配置グループは、使用する冗長性アルゴリズムに基づいて構築されます。3 方向のレプリケーションでは、冗長性は 3 つの異なる OSD を使用するように設定されます。イレイジャーコーディングされたプールの場合、使用する OSD の数はチャンクの数で定義されます。

プールを定義する場合、配置グループの数によって、使用可能なすべての OSD にデータが分散される粒度のグレードが定義されます。数値が大きいほど、容量負荷の均等化が向上します。ただし、データを再構築する場合は配置グループの処理も重要であるため、事前に慎重に選択する必要があります。計算をサポートするには、アジャイル環境の生成に利用できるツールを使用できます。

ストレージクラスターの有効期間中、プールが最初に予想される制限を上回る可能性があります。ドライブの数が増えると、再計算が推奨されます。OSD ごとの配置グループの数は約 100 である必要があります。ストレージクラスターに OSD を追加すると、OSD あたりの PG の数は時間の経過とともに減少します。ストレージクラスターで最初に 120 ドライブを使用し、プールの **pg_num** を 4000 に設定すると、レプリケーション係数が 3 の場合に、OSD ごとに 100 PG に設定されます。時間の経過とともに、OSD の数が 10 倍になると、OSD あたりの PG の数は 10 になります。OSD ごとの PG の数が少ないと、容量が不均一に分散される傾向があるため、プールごとの PG を調整することを検討してください。

配置グループ数の調整は、オンラインで実行できます。再計算は PG 番号の再計算だけでなく、データの再配置が関係し、長いプロセスになります。ただし、データの可用性はいつでも維持されます。

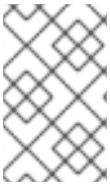
OSD ごとの非常に高い PG は、失敗した OSD 上でのすべての PG を再構築すると同時に起動されるため、回避する必要があります。時間内に再構築を実行するには、多くの IOPS が必要ですが、これは利用できない可能性があります。これにより、I/O キューが深くなり、待ち時間が長くなり、ストレージクラスターが使用できなくなったり、修復時間が長くなったりします。

関連情報

- 特定のユースケースで値を計算する場合は、PG calculator を参照してください。
- 詳細は、**Red Hat Ceph Storage ストラテジーガイド** の **イレイジャーコードプール** の章を参照してください。

第7章 CEPH ORCHESTRATOR を使用したモニタリングスタックの管理

ストレージ管理者は、バックエンドにて Cephadm と Ceph Orchestrator を使用して、モニタリングおよびアラートスタックをデプロイできます。モニタリングスタックは、Prometheus、Prometheus エクスポーター、Prometheus Alertmanager、および Grafana で設定されます。ユーザーは、YAML 設定ファイルの Cephadm でこれらのサービスを定義するか、コマンドラインインターフェイスを使用してそれらをデプロイする必要があります。同じタイプの複数のサービスがデプロイされると、可用性の高い設定がデプロイされます。ノードエクスポーターはこのルールの例外です。



注記

Red Hat Ceph Storage 6.0 は、Prometheus、Grafana、Alertmanager、および node-exporter などのモニタリングサービスをデプロイするためのカスタムイメージをサポートしません。

以下のモニタリングサービスは、Cephadm でデプロイすることができます。

- Prometheus はモニタリングおよびアラートツールキットです。これは、Prometheus エクスポーターによって提供されるデータを収集し、事前に定義されたしきい値に達した場合に事前設定されたアラートを実行します。Prometheus マネージャーモジュールは、**ceph-mgr** のコレクションポイントから Ceph パフォーマンスカウンターに渡す Prometheus エクスポーターを提供します。
  デーモンを提供するメトリックなど、スクレープターゲットなどの Prometheus 設定は、Cephadm によって自動的に設定されます。Cephadm は、デフォルトのアラートのリスト (例: health error、10% OSDs down、pgs inactive) もデプロイします。

- Alertmanager は、Prometheus サーバーによって送信されるアラートを処理します。これは、正しい受信側にアラートを複製解除、グループ化、およびルーティングします。デフォルトでは、Ceph Dashboard は受信側として自動的に設定されます。Alertmanager は、Prometheus サーバーによって送信されるアラートを処理します。アラートは Alertmanager を使用して無音にすることができますが、無音は Ceph Dashboard を使用して管理することもできます。

- Grafana は視覚化および警告ソフトウェアです。Grafana のアラート機能は、このモニタリングスタックによって使用されません。アラートについては、Alertmanager が使用されます。
  デフォルトでは、Grafana へのトラフィックは TLS で暗号化されます。独自の TLS 証明書を指定するか、自己署名の証明書を使用できます。Grafana をデプロイする前にカスタム証明書が設定されていない場合、自己署名証明書が自動的に作成され、Grafana に設定されます。
  Grafana のカスタム証明書は、以下のコマンドを使用して設定できます。

  構文

  ```
  ceph config-key set mgr/cephadm/grafana_key -i
  PRESENT_WORKING_DIRECTORY/key.pem
  ceph config-key set mgr/cephadm/grafana_crt -i
  PRESENT_WORKING_DIRECTORY/certificate.pem
  ```

ノードエクスポーターは Prometheus のエクスポーターで、インストールされているノードに関するデータを提供します。ノードエクスポーターをすべてのノードにインストールすることが推奨されます。これは、node-exporter サービスタイプの **monitoring.yml** ファイルを使用して実行できます。

7.1. CEPH ORCHESTRATOR を使用したモニタリングスタックのデプロイ

モニタリングスタックは、Prometheus、Prometheus エクスポーター、Prometheus Alertmanager、および Grafana で設定されます。Ceph Dashboard はこれらのコンポーネントを使用して、クラスターの使用状況およびパフォーマンスの詳細なメトリックを保存し、可視化します。

YAML ファイル形式のサービス仕様を使用して、モニタリングスタックをデプロイできます。すべてのモニタリングサービスのバインド先のネットワークおよびポートは **yml** ファイルで設定できます。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- ノードへの root レベルのアクセス。

手順

1. Ceph Manager デーモンで prometheus モジュールを有効にします。これにより、内部 Ceph メトリックが公開され、Prometheus がそれらを読み取りできます。

   例

   ```
   [ceph: root@host01 /]# ceph mgr module enable prometheus
   ```

   重要

   このコマンドは、Prometheus のデプロイ前に実行されるようにします。デプロイメントの前にコマンドが実行されなかった場合は、Prometheus を再デプロイして設定を更新する必要があります。

   ```
   ceph orch redeploy prometheus
   ```

2. 以下のディレクトリーに移動します。

   構文

   ```
   cd /var/lib/ceph/DAEMON_PATH/
   ```

   例

   ```
   [ceph: root@host01 mds/]# cd /var/lib/ceph/monitoring/
   ```

   注記

   ディレクトリー **monitoring** が存在しない場合は、作成します。

3. **monitoring.yml** ファイルを作成します。

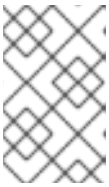
   例

   ```
   [ceph: root@host01 monitoring]# touch monitoring.yml
   ```

4. 以下の例のような内容で仕様ファイルを編集します。

例

```
service_type: prometheus
service_name: prometheus
placement:
  hosts:
  - host01
networks:
- 192.169.142.0/24
---
service_type: node-exporter
---
service_type: alertmanager
service_name: alertmanager
placement:
  hosts:
  - host01
networks:
- 192.169.142.0/24
---
service_type: grafana
service_name: grafana
placement:
  hosts:
  - host01
networks:
- 192.169.142.0/24
```



注記

監視スタックコンポーネントの **alertmanager**、**prometheus**、および **grafana** が同じホストにデプロイされていることを確認します。**node-exporter** コンポーネントはすべてのホストにデプロイする必要があります。

5. モニタリングサービスを適用します。

例

```
[ceph: root@host01 monitoring]# ceph orch apply -i monitoring.yml
```

検証

- サービスをリスト表示します。

例

```
[ceph: root@host01 /]# ceph orch ls
```

- ホスト、デーモン、およびプロセスをリスト表示します。

構文

```
ceph orch ps --service_name=SERVICE_NAME
```

例

```
[ceph: root@host01 /]# ceph orch ps --service_name=prometheus
```

重要

Prometheus、Grafana、および Ceph Dashboard はすべて相互に対話するように自動設定されるため、Ceph Dashboard で Grafana 統合が完全に機能します。

7.2. CEPH ORCHESTRATOR を使用したモニタリングスタックの削除

ceph orch rm コマンドを使用してモニタリングスタックを削除できます。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。

手順

1. Cephadm シェルにログインします。

   例

   ```
   [root@host01 ~]# cephadm shell
   ```

2. **ceph orch rm** コマンドを使用してモニタリングスタックを削除します。

   構文

   ```
   ceph orch rm SERVICE_NAME --force
   ```

   例

   ```
   [ceph: root@host01 /]# ceph orch rm grafana
   [ceph: root@host01 /]# ceph orch rm prometheus
   [ceph: root@host01 /]# ceph orch rm node-exporter
   [ceph: root@host01 /]# ceph orch rm alertmanager
   [ceph: root@host01 /]# ceph mgr module disable prometheus
   ```

3. プロセスのステータスを確認します。

   例

   ```
   [ceph: root@host01 /]# ceph orch status
   ```

検証

- サービスをリスト表示します。

  例

```
[ceph: root@host01 /]# ceph orch ls
```

- ホスト、デーモン、およびプロセスをリスト表示します。

  構文

  ```
  ceph orch ps
  ```

  例

  ```
  [ceph: root@host01 /]# ceph orch ps
  ```

関連情報

- 詳細は、**Red Hat Ceph Storage オペレーションガイド** の Ceph Orchestrator を使用したデプロイメント セクションを参照してください。

第8章 基本的な RED HAT CEPH STORAGE のクライアント設定

ストレージ管理者は、ストレージクラスターと対話するために基本設定でクライアントマシンを設定する必要があります。ほとんどのクライアントマシンには **ceph-common package** とその依存関係のみがインストールされている必要があります。これは、基本的な **ceph** コマンドおよび **rados** コマンドと、**mount.ceph** や **rbd** などのその他のコマンドを提供します。

8.1. クライアントマシンでのファイルの設定

通常、クライアントマシンには完全なストレージクラスターメンバーよりも小さな設定ファイルが必要になります。Ceph モニターに到達するためにクライアントに情報を提供する最小限の設定ファイルを生成できます。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- ノードへの root アクセス。

手順

1. ファイルを設定するノードで、**/etc** フォルダーにディレクトリー **ceph** を作成します。

   例

   ```
   [root@host01 ~]# mkdir /etc/ceph/
   ```

2. **/etc/ceph** ディレクトリーに移動します。

   例

   ```
   [root@host01 ~]# cd /etc/ceph/
   ```

3. **ceph** ディレクトリーで設定ファイルを生成します。

   例

   ```
   [root@host01 ceph]# ceph config generate-minimal-conf

   # minimal ceph.conf for 417b1d7a-a0e6-11eb-b940-001a4a000740
   [global]
    fsid = 417b1d7a-a0e6-11eb-b940-001a4a000740
    mon_host = [v2:10.74.249.41:3300/0,v1:10.74.249.41:6789/0]
   ```

   このファイルのコンテンツは、**/etc/ceph/ceph.conf** パスにインストールする必要があります。この設定ファイルを使用して、Ceph モニターに到達できます。

8.2. クライアントマシンでのキーリングの設定

ほとんどの Ceph クラスターは認証が有効な状態で実行され、クライアントはクラスターマシンと通信するために鍵が必要です。Ceph モニターに到達するためにクライアントに詳細を提供できるキーリングを生成できます。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- ノードへの root アクセス。

手順

1. キーリングを設定するノードで、**/etc** フォルダーにディレクトリー **ceph** を作成します。

   例

   ```
   [root@host01 ~]# mkdir /etc/ceph/
   ```

2. **ceph** ディレクトリーの **/etc/ceph** ディレクトリーに移動します。

   例

   ```
   [root@host01 ~]# cd /etc/ceph/
   ```

3. クライアントのキーリングを生成します。

   構文

   ```
   ceph auth get-or-create client.CLIENT_NAME -o /etc/ceph/NAME_OF_THE_FILE
   ```

   例

   ```
   [root@host01 ceph]# ceph auth get-or-create client.fs -o /etc/ceph/ceph.keyring
   ```

4. **ceph.keyring** ファイルの出力を確認します。

   例

   ```
   [root@host01 ceph]# cat ceph.keyring

   [client.fs]
    key = AQAvoH5gkUCsExAATz3xCBLd4n6B6jRv+Z7CVQ==
   ```

   出力は **/etc/ceph/ceph.keyring** などのキーリングファイルに配置する必要があります。

第9章 CEPH ORCHESTRATOR を使用した MDS サービスの管理

ストレージ管理者は、バックエンドにて Cephadm と Ceph Orchestrator を使用して MDS サービスをデプロイできます。デフォルトでは、Ceph File System (CephFS) はアクティブな MDS デーモンを 1 つだけ使用します。ただし、多くのクライアントがあるシステムでは複数のアクティブな MDS デーモンを使用する利点があります。

本セクションでは、以下の管理タスクを説明します。

- コマンドラインインターフェイスを使用した MDS サービスのデプロイ。
- サービス仕様を使用した MDS サービスのデプロイ。
- Ceph Orchestrator を使用した MDS サービスの削除。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- すべてのノードへの root レベルのアクセス。
- ホストがクラスターに追加されている。
- すべてのマネージャー、モニター、および OSD デーモンがデプロイされている。

9.1. コマンドラインインターフェイスを使用した MDS サービスのデプロイ

Ceph Orchestrator を使用すると、コマンドラインインターフェイスで **placement** 仕様を使用して、Metadata Server (MDS) サービスをデプロイできます。Ceph ファイルシステム (CephFS) には、1 つ以上の MDS が必要です。

注記

最低でも、Ceph ファイルシステム (CephFS) データ用のプール 1 つと CephFS メタデータ用のプール 1 つの 2 つのプールがあるようにしてください。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- ホストがクラスターに追加されている。
- すべてのマネージャー、モニター、および OSD デーモンがデプロイされます。

手順

1. Cephadm シェルにログインします。

   例

   ```
   [root@host01 ~]# cephadm shell
   ```

2. 配置仕様を使用して MDS デーモンをデプロイする方法は 2 つあります。

方法 1

- **ceph fs volume** を使用して MDS デーモンを作成します。これにより、CephFS に関連付けられた CephFS ボリュームとプールが作成され、ホストで MDS サービスも開始されます。

  構文

  ```
  ceph fs volume create FILESYSTEM_NAME --placement="NUMBER_OF_DAEMONS HOST_NAME_1 HOST_NAME_2 HOST_NAME_3"
  ```

  注記

  デフォルトでは、このコマンドに対してレプリケートされたプールが作成されます。

  例

  ```
  [ceph: root@host01 /]# ceph fs volume create test --placement="2 host01 host02"
  ```

方法 2

- プール、CephFS を作成してから、配置仕様を使用して MDS サービスをデプロイします。

  a. CephFS のプールを作成します。

  構文

  ```
  ceph osd pool create DATA_POOL [PG_NUM]
  ceph osd pool create METADATA_POOL [PG_NUM]
  ```

  例

  ```
  [ceph: root@host01 /]# ceph osd pool create cephfs_data 64
  [ceph: root@host01 /]# ceph osd pool create cephfs_metadata 64
  ```

  通常、メタデータプールは、データプールよりもオブジェクトがはるかに少ないため、控えめな数の配置グループ (PG) で開始できます。必要に応じて PG の数を増やすことができます。プールサイズの範囲は 64 PG ~ 512 PG です。データプールのサイズは、ファイルシステム内で予想されるファイルの数とサイズに比例します。

  重要

  メタデータプールでは、以下を使用することを検討してください。

  - このプールへのデータ損失によりファイルシステム全体にアクセスできなくなる可能性があるため、レプリケーションレベルが高くなります。
  - Solid-State Drive (SSD) ディスクなどのレイテンシーが低くなるストレージ。これは、クライアントで観察されるファイルシステム操作のレイテンシーに直接影響するためです。

  b. データプールおよびメタデータプールのファイルシステムを作成します。

構文

```
ceph fs new FILESYSTEM_NAME METADATA_POOL DATA_POOL
```

例

```
[ceph: root@host01 /]# ceph fs new test cephfs_metadata cephfs_data
```

c. **ceph orch apply** コマンドを使用して MDS サービスをデプロイします。

構文

```
ceph orch apply mds FILESYSTEM_NAME --placement="NUMBER_OF_DAEMONS HOST_NAME_1 HOST_NAME_2 HOST_NAME_3"
```

例

```
[ceph: root@host01 /]# ceph orch apply mds test --placement="2 host01 host02"
```

検証

- サービスをリスト表示します。

  例

  ```
  [ceph: root@host01 /]# ceph orch ls
  ```

- CephFS のステータスを確認します。

  例

  ```
  [ceph: root@host01 /]# ceph fs ls
  [ceph: root@host01 /]# ceph fs status
  ```

- ホスト、デーモン、およびプロセスをリスト表示します。

  構文

  ```
  ceph orch ps --daemon_type=DAEMON_NAME
  ```

  例

  ```
  [ceph: root@host01 /]# ceph orch ps --daemon_type=mds
  ```

関連情報

- Ceph ファイルシステム (CephFS) の作成に関する詳細は、**Red Hat Ceph Storage ファイルシステムガイド** を参照してください。
- プール値の設定の詳細は、プール内の配置グループ数の設定 を参照してください。

9.2. サービス仕様を使用した MDS サービスのデプロイ

Ceph Orchestrator を使用すると、サービス仕様を使用して MDS サービスをデプロイできます。

注記

少なくとも 2 つのプールがあることを確認してください。1 つは Ceph ファイルシステム (CephFS) データ用で、もう 1 つは CephFS メタデータ用です。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- ホストがクラスターに追加されている。
- すべてのマネージャー、モニター、および OSD デーモンがデプロイされます。

手順

1. **mds.yaml** ファイルを作成します。

   例

   ```
   [root@host01 ~]# touch mds.yaml
   ```

2. **mds.yaml** ファイルを編集し、以下の詳細を含めます。

   構文

   ```
   service_type: mds
   service_id: FILESYSTEM_NAME
   placement:
     hosts:
     - HOST_NAME_1
     - HOST_NAME_2
     - HOST_NAME_3
   ```

   例

   ```
   service_type: mds
   service_id: fs_name
   placement:
     hosts:
     - host01
     - host02
   ```

3. YAML ファイルをコンテナー内のディレクトリーにマウントします。

   例

   ```
   [root@host01 ~]# cephadm shell --mount mds.yaml:/var/lib/ceph/mds/mds.yaml
   ```

4. そのディレクトリーに移動します。

例

```
[ceph: root@host01 /]# cd /var/lib/ceph/mds/
```

5. Cephadm シェルにログインします。

例

```
[root@host01 ~]# cephadm shell
```

6. 以下のディレクトリーに移動します。

例

```
[ceph: root@host01 /]# cd /var/lib/ceph/mds/
```

7. サービス仕様を使用して MDS サービスをデプロイします。

構文

```
ceph orch apply -i FILE_NAME.yaml
```

例

```
[ceph: root@host01 mds]# ceph orch apply -i mds.yaml
```

8. MDS サービスがデプロイされ、機能したら、CephFS を作成します。

構文

```
ceph fs new CEPHFS_NAME METADATA_POOL DATA_POOL
```

例

```
[ceph: root@host01 /]# ceph fs new test metadata_pool data_pool
```

検証

- サービスをリスト表示します。

例

```
[ceph: root@host01 /]# ceph orch ls
```

- ホスト、デーモン、およびプロセスをリスト表示します。

構文

```
ceph orch ps --daemon_type=DAEMON_NAME
```

例

```
[ceph: root@host01 /]# ceph orch ps --daemon_type=mds
```

関連情報

- Ceph ファイルシステム (CephFS) の作成に関する詳細は、Red Hat Ceph Storage ファイルシステムガイド を参照してください。

9.3. CEPH ORCHESTRATOR を使用した MDS サービスの削除

ceph orch rm コマンドを使用してサービスを削除できます。または、ファイルシステムおよび関連するプールを削除できます。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- すべてのノードへの root レベルのアクセス。
- ホストがクラスターに追加されている。
- ホストにデプロイされた MDS デーモン 1 つ以上。

手順

- MDS デーモンをクラスターから削除する方法は 2 つあります。

方法 1

- CephFS ボリューム、関連するプール、およびサービスを削除します。
  a. Cephadm シェルにログインします。

  例

  ```
  [root@host01 ~]# cephadm shell
  ```

  b. 設定パラメーター **mon_allow_pool_delete** を **true** に設定します。

  例

  ```
  [ceph: root@host01 /]# ceph config set mon mon_allow_pool_delete true
  ```

  c. ファイルシステムを削除します。

  構文

  ```
  ceph fs volume rm FILESYSTEM_NAME --yes-i-really-mean-it
  ```

  例

```
[ceph: root@host01 /]# ceph fs volume rm cephfs-new --yes-i-really-mean-it
```

このコマンドは、ファイルシステム、そのデータ、メタデータプールを削除します。また、有効な **ceph-mgr** Orchestrator モジュールを使用して MDS を削除しようとします。

方法 2

- **ceph orch rm** コマンドを使用して、クラスター全体から MDS サービスを削除します。

  a. サービスをリスト表示します。

  例

  ```
  [ceph: root@host01 /]# ceph orch ls
  ```

  b. サービスの削除

  構文

  ```
  ceph orch rm SERVICE_NAME
  ```

  例

  ```
  [ceph: root@host01 /]# ceph orch rm mds.test
  ```

検証

- ホスト、デーモン、およびプロセスをリスト表示します。

  構文

  ```
  ceph orch ps
  ```

  例

  ```
  [ceph: root@host01 /]# ceph orch ps
  ```

関連情報

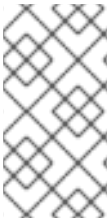
- 詳細は、**Red Hat Ceph Storage オペレーションガイド** の Deploying the MDS service using the command line interface セクションを参照してください。
- 詳細は、**Red Hat Ceph Storage オペレーションガイド**の サービス仕様を使用した MDS サービスのデプロイメント セクションを参照してください。

第10章 CEPH ORCHESTRATOR を使用した CEPH オブジェクトゲートウェイの管理

ストレージ管理者は、コマンドラインインターフェイスまたはサービス仕様を使用して、Ceph オブジェクトゲートウェイをデプロイできます。

また、マルチサイトオブジェクトゲートウェイを設定し、Ceph Orchestrator を使用して Ceph オブジェクトゲートウェイを削除することもできます。

Cephadm は、Ceph オブジェクトゲートウェイを、単一クラスターデプロイメントを管理するデーモンのコレクションまたはマルチサイトデプロイメントの特定のレルムおよびゾーンを管理するデーモンのコレクションとしてデプロイします。



注記

Cephadm では、オブジェクトゲートウェイデーモンは、**ceph.conf** やコマンドラインではなく、モニター設定データベースを使用して設定されます。その設定がまだ **client.rgw** セクションにない場合、オブジェクトゲートウェイデーモンはデフォルト設定で起動し、ポート **80** にバインドします。

注記

.default.rgw.buckets.index プールは、Ceph Object Gateway でバケットが作成された後にのみ作成されます。一方、データはバケットにアップロードされた後に **.default.rgw.buckets.data** プールが作成されます。

本セクションでは、以下の管理タスクを説明します。

- コマンドラインインターフェイスを使用した Ceph オブジェクトゲートウェイのデプロイ。
- サービス仕様を使用した Ceph オブジェクトゲートウェイのデプロイ。
- Ceph Orchestrator を使用したマルチサイト Ceph Object Gateway のデプロイ
- Ceph Orchestrator を使用した Ceph オブジェクトゲートウェイの削除

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- すべてのノードへの root レベルのアクセス。
- ホストがクラスターに追加されている。
- すべてのマネージャー、モニター、および OSD がストレージクラスターにデプロイされます。

10.1. コマンドラインインターフェイスを使用した CEPH オブジェクトゲートウェイのデプロイ

Ceph Orchestrator を使用すると、コマンドラインインターフェイスで **ceph orch** コマンドを使用して Ceph オブジェクトゲートウェイをデプロイできます。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- すべてのノードへの root レベルのアクセス。
- ホストがクラスターに追加されている。
- すべてのマネージャー、モニター、および OSD デーモンがデプロイされている。

手順

1. Cephadm シェルにログインします。

   例

   ```
   [root@host01 ~]# cephadm shell
   ```

2. Ceph オブジェクトゲートウェイデーモンは、以下の 3 つの方法でデプロイできます。

方法 1

- レルム、ゾーングループ、ゾーンを作成し、ホスト名で配置仕様を使用します。

  a. レルムを作成します。

  構文

  ```
  radosgw-admin realm create --rgw-realm=REALM_NAME --default
  ```

  例

  ```
  [ceph: root@host01 /]# radosgw-admin realm create --rgw-realm=test_realm --default
  ```

  b. ゾーングループを作成します。

  構文

  ```
  radosgw-admin zonegroup create --rgw-zonegroup=ZONE_GROUP_NAME  --master --default
  ```

  例

  ```
  [ceph: root@host01 /]# radosgw-admin zonegroup create --rgw-zonegroup=default  --master --default
  ```

  c. ゾーンを作成します。

  構文

  ```
  radosgw-admin zone create --rgw-zonegroup=ZONE_GROUP_NAME --rgw-zone=ZONE_NAME --master --default
  ```

  例

  –

```
[ceph: root@host01 /]# radosgw-admin zone create --rgw-zonegroup=default --rgw-zone=test_zone --master --default
```

d. 変更をコミットします。

構文

```
radosgw-admin period update --rgw-realm=REALM_NAME --commit
```

例

```
[ceph: root@host01 /]# radosgw-admin period update --rgw-realm=test_realm --commit
```

e. **ceph orch apply** コマンドを実行します。

構文

```
ceph orch apply rgw NAME [--realm=REALM_NAME] [--zone=ZONE_NAME] --placement="NUMBER_OF_DAEMONS [HOST_NAME_1 HOST_NAME_2]"
```

例

```
[ceph: root@host01 /]# ceph orch apply rgw test --realm=test_realm --zone=test_zone --placement="2 host01 host02"
```

方法 2

- 任意のサービス名を使用して、単一のクラスターデプロイメント用に 2 つの Ceph オブジェクトゲートウェイデーモンをデプロイします。

構文

```
ceph orch apply rgw SERVICE_NAME
```

例

```
[ceph: root@host01 /]# ceph orch apply rgw foo
```

方法 3

- ホストのラベル付きセットで任意のサービス名を使用します。

構文

```
ceph orch host label add HOST_NAME_1 LABEL_NAME
ceph orch host label add HOSTNAME_2 LABEL_NAME
ceph orch apply rgw SERVICE_NAME --placement="label:LABEL_NAME count-per-host:NUMBER_OF_DAEMONS" --port=8000
```

注記

NUMBER_OF_DAEMONS は、各ホストにデプロイされる Ceph オブジェクトゲートウェイの数を制御します。追加のコストを発生させずに最高のパフォーマンスを実現するには、この値を 2 に設定します。

例

```
[ceph: root@host01 /]# ceph orch host label add host01 rgw  # the 'rgw' label can be anything
[ceph: root@host01 /]# ceph orch host label add host02 rgw
[ceph: root@host01 /]# ceph orch apply rgw foo --placement="2 label:rgw" --port=8000
```

検証

- サービスをリスト表示します。

  例

  ```
  [ceph: root@host01 /]# ceph orch ls
  ```

- ホスト、デーモン、およびプロセスをリスト表示します。

  構文

  ```
  ceph orch ps --daemon_type=DAEMON_NAME
  ```

  例

  ```
  [ceph: root@host01 /]# ceph orch ps --daemon_type=rgw
  ```

10.2. サービス仕様を使用した CEPH OBJECT GATEWAY のデプロイ

Ceph Object Gateway は、デフォルトまたはカスタムのレルム、ゾーン、およびゾーングループいずれかのサービス仕様を使用してデプロイできます。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- ブートストラップされたホストへの root レベルのアクセス。
- ホストがクラスターに追加されている。
- すべてのマネージャー、モニター、および OSD デーモンがデプロイされます。

手順

1. root ユーザーとして、仕様ファイルを作成します。

   例

   ```
   [root@host01 ~]# touch radosgw.yml
   ```

2. **radosgw.yml** ファイルを編集し、デフォルトレルム、ゾーン、およびゾーングループの以下の詳細を追加します。

構文

```
service_type: rgw
service_id: REALM_NAME.ZONE_NAME
placement:
  hosts:
  - HOST_NAME_1
  - HOST_NAME_2
  count_per_host: NUMBER_OF_DAEMONS
spec:
  rgw_realm: REALM_NAME
  rgw_zone: ZONE_NAME
  rgw_frontend_port: FRONT_END_PORT
networks:
  -  NETWORK_CIDR # Ceph Object Gateway service binds to a specific network
```

注記

NUMBER_OF_DAEMONS は、各ホストにデプロイされる Ceph Object Gateway の数を制御します。追加のコストを発生させずに最高のパフォーマンスを実現するには、この値を 2 に設定します。

例

```
service_type: rgw
service_id: default
placement:
  hosts:
  - host01
  - host02
  - host03
  count_per_host: 2
spec:
  rgw_realm: default
  rgw_zone: default
  rgw_frontend_port: 1234
networks:
  - 192.169.142.0/24
```

3. オプション: カスタムレルム、ゾーン、およびゾーングループの場合は、リソースを作成し、**radosgw.yml** ファイルを作成します。

   a. カスタムレルム、ゾーン、およびゾーングループを作成します。

   例

```
[root@host01 ~]# radosgw-admin realm create --rgw-realm=test_realm
[root@host01 ~]# radosgw-admin zonegroup create --rgw-zonegroup=test_zonegroup
[root@host01 ~]# radosgw-admin zone create --rgw-zonegroup=test_zonegroup --rgw-zone=test_zone
[root@host01 ~]# radosgw-admin period update --rgw-realm=test_realm --commit
```

b. 以下の内容で **radosgw.yml** ファイルを作成します。

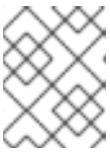
例

```
service_type: rgw
service_id: test_realm.test_zone
placement:
  hosts:
  - host01
  - host02
  - host03
  count_per_host: 2
spec:
  rgw_realm: test_realm
  rgw_zone: test_zone
  rgw_frontend_port: 1234
networks:
  - 192.169.142.0/24
```

4. **radosgw.yml** ファイルをコンテナーのディレクトリーにマウントします。

例

```
[root@host01 ~]# cephadm shell --mount radosgw.yml:/var/lib/ceph/radosgw/radosgw.yml
```



注記

シェルを終了するたびに、デーモンをデプロイする前にファイルをコンテナーにマウントする必要があります。

5. サービス仕様を使用して Ceph Object Gateway をデプロイします。

構文

```
ceph orch apply -i FILE_NAME.yml
```

例

```
[ceph: root@host01 /]# ceph orch apply -i radosgw.yml
```

検証

- サービスをリスト表示します。

例

```
[ceph: root@host01 /]# ceph orch ls
```

- ホスト、デーモン、およびプロセスをリスト表示します。

構文

```
ceph orch ps --daemon_type=DAEMON_NAME
```

例

```
[ceph: root@host01 /]# ceph orch ps --daemon_type=rgw
```

10.3. CEPH ORCHESTRATOR を使用したマルチサイト CEPH OBJECT GATEWAY のデプロイ

Ceph Orchestrator は、Ceph Object Gateway のマルチサイト設定オプションをサポートしています。

各オブジェクトゲートウェイを active-active ゾーン設定で機能するように設定すると、非プライマリーゾーンへの書き込みが可能になります。マルチサイト設定は、レルムと呼ばれるコンテナー内に保存されます。

レルムは、ゾーングループ、ゾーン、および期間を保存します。**rgw** デーモンは同期を処理し、個別の同期エージェントが不要になるため、アクティブ-アクティブ設定で動作します。

コマンドラインインターフェイス (CLI) を使用してマルチサイトゾーンをデプロイすることもできます。

注記

以下の設定では、少なくとも 2 つの Red Hat Ceph Storage クラスターが地理的に別々の場所であることを前提としています。ただし、この設定は同じサイトでも機能します。

前提条件

- 少なくとも 2 つの実行中の Red Hat Ceph Storage クラスター
- 少なくとも 2 つの Ceph Object Gateway インスタンス (各 Red Hat Ceph Storage クラスターに 1 つ)。
- すべてのノードへの root レベルのアクセス。
- ノードまたはコンテナーがストレージクラスターに追加されます。
- すべての Ceph Manager、Monitor、および OSD デーモンがデプロイされます。

手順

1. **cephadm** シェルで、プライマリーゾーンを設定します。
   a. レルムを作成します。

   構文

   ```
   radosgw-admin realm create --rgw-realm=REALM_NAME --default
   ```

   例

   ```
   [ceph: root@host01 /]# radosgw-admin realm create --rgw-realm=test_realm --default
   ```

ストレージクラスターに単一のレルムがある場合は、**--default** フラグを指定します。

b. プライマリーゾーングループを作成します。

構文

```
radosgw-admin zonegroup create --rgw-zonegroup=ZONE_GROUP_NAME --endpoints=http://RGW_PRIMARY_HOSTNAME:RGW_PRIMARY_PORT_NUMBER_1 --master --default
```

例

```
[ceph: root@host01 /]# radosgw-admin zonegroup create --rgw-zonegroup=us --endpoints=http://rgw1:80 --master --default
```

c. プライマリーゾーンを作成します。

構文

```
radosgw-admin zone create --rgw-zonegroup=PRIMARY_ZONE_GROUP_NAME --rgw-zone=PRIMARY_ZONE_NAME --endpoints=http://RGW_PRIMARY_HOSTNAME:RGW_PRIMARY_PORT_NUMBER_1 --access-key=SYSTEM_ACCESS_KEY --secret=SYSTEM_SECRET_KEY
```

例

```
[ceph: root@host01 /]# radosgw-admin zone create --rgw-zonegroup=us --rgw-zone=us-east-1 --endpoints=http://rgw1:80 --access-key=LIPEYZJLTWXRKXS9LPJC --secret-key=IsAje0AVDNXNw48LjMAimpCpI7VaxJYSnfD0FFKQ
```

d. 必要に応じて、デフォルトゾーン、ゾーングループ、および関連するプールを削除します。

重要

デフォルトゾーンおよびゾーングループを使用してデータを保存する場合は、デフォルトゾーンとそのプールは削除しないでください。また、デフォルトのゾーングループを削除して、システムユーザーを削除します。

default ゾーンおよびゾーングループの古いデータにアクセスするには、**radosgw-admin** コマンドで **--rgw-zone default** および **--rgw-zonegroup default** を使用します。

例

```
[ceph: root@host01 /]# radosgw-admin zonegroup delete --rgw-zonegroup=default
[ceph: root@host01 /]# ceph osd pool rm default.rgw.log default.rgw.log --yes-i-really-really-mean-it
[ceph: root@host01 /]# ceph osd pool rm default.rgw.meta default.rgw.meta --yes-i-really-really-mean-it
[ceph: root@host01 /]# ceph osd pool rm default.rgw.control default.rgw.control --yes-i-really-really-mean-it
[ceph: root@host01 /]# ceph osd pool rm default.rgw.data.root default.rgw.data.root --
```

```
yes-i-really-really-mean-it
[ceph: root@host01 /]# ceph osd pool rm default.rgw.gc default.rgw.gc --yes-i-really-really-mean-it
```

e. システムユーザーを作成します。

構文

```
radosgw-admin user create --uid=USER_NAME --display-name="USER_NAME" --access-key=SYSTEM_ACCESS_KEY --secret=SYSTEM_SECRET_KEY --system
```

例

```
[ceph: root@host01 /]# radosgw-admin user create --uid=zone.user --display-name="Zone user" --system
```

access_key および **secret_key** を書き留めておきます。

f. アクセスキーとシステムキーをプライマリーゾーンに追加します。

構文

```
radosgw-admin zone modify --rgw-zone=PRIMARY_ZONE_NAME --access-key=ACCESS_KEY --secret=SECRET_KEY
```

例

```
[ceph: root@host01 /]# radosgw-admin zone modify --rgw-zone=us-east-1 --access-key=NE48APYCAODEPLKBCZVQ--secret=u24GHQWRE3yxxNBnFBzjM4jn14mFIckQ4EKL6LoW
```

g. 変更をコミットします。

構文

```
radosgw-admin period update --commit
```

例

```
[ceph: root@host01 /]# radosgw-admin period update --commit
```

h. **cephadm** シェルの外部で、ストレージクラスターおよびプロセスの **FSID** を取得します。

例

```
[root@host01 ~]#  systemctl list-units | grep ceph
```

i. Ceph Object Gateway デーモンを起動します。

構文

```
systemctl start ceph-FSID@DAEMON_NAME
systemctl enable ceph-FSID@DAEMON_NAME
```

例

```
[root@host01 ~]# systemctl start ceph-62a081a6-88aa-11eb-a367-
001a4a000672@rgw.test_realm.us-east-1.host01.ahdtsw.service
[root@host01 ~]# systemctl enable ceph-62a081a6-88aa-11eb-a367-
001a4a000672@rgw.test_realm.us-east-1.host01.ahdtsw.service
```

2. Cephadm シェルで、セカンダリーゾーンを設定します。

   a. ホストからプライマリーレルム設定をプルします。

   構文

   ```
   radosgw-admin realm pull --url=URL_TO_PRIMARY_ZONE_GATEWAY --access-
   key=ACCESS_KEY --secret-key=SECRET_KEY
   ```

   例

   ```
   [ceph: root@host04 /]# radosgw-admin realm pull --url=http://10.74.249.26:80 --access-
   key=LIPEYZJLTWXRKXS9LPJC --secret-
   key=IsAje0AVDNXNw48LjMAimpCpI7VaxJYSnfD0FFKQ
   ```

   b. ホストからプライマリー期間設定をプルします。

   構文

   ```
   radosgw-admin period pull --url=URL_TO_PRIMARY_ZONE_GATEWAY --access-
   key=ACCESS_KEY --secret-key=SECRET_KEY
   ```

   例

   ```
   [ceph: root@host04 /]# radosgw-admin period pull --url=http://10.74.249.26:80 --access-
   key=LIPEYZJLTWXRKXS9LPJC --secret-
   key=IsAje0AVDNXNw48LjMAimpCpI7VaxJYSnfD0FFKQ
   ```

   c. セカンダリーゾーンを設定します。

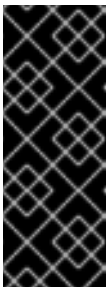
   構文

   ```
   radosgw-admin zone create --rgw-zonegroup=ZONE_GROUP_NAME \
            --rgw-zone=SECONDARY_ZONE_NAME --
   endpoints=http://RGW_SECONDARY_HOSTNAME:RGW_PRIMARY_PORT_NUMBER
   _1 \
            --access-key=SYSTEM_ACCESS_KEY --secret=SYSTEM_SECRET_KEY \
            --endpoints=http://FQDN:80 \
            [--read-only]
   ```

   例

```
[ceph: root@host04 /]# radosgw-admin zone create --rgw-zonegroup=us --rgw-zone=us-east-2 --endpoints=http://rgw2:80 --access-key=LIPEYZJLTWXRKXS9LPJC --secret-key=IsAje0AVDNXNw48LjMAimpCpI7VaxJYSnfD0FFKQ --endpoints=http://rgw.example.com:80
```

d. 必要に応じて、デフォルトゾーンを削除します。



重要

デフォルトゾーンおよびゾーングループを使用してデータを保存する場合は、デフォルトゾーンとそのプールは削除しないでください。

default ゾーンおよびゾーングループの古いデータにアクセスするには、**radosgw-admin** コマンドで **--rgw-zone default** および **--rgw-zonegroup default** を使用します。

例

```
[ceph: root@host04 /]# radosgw-admin zone rm --rgw-zone=default
[ceph: root@host04 /]# ceph osd pool rm default.rgw.log default.rgw.log --yes-i-really-really-mean-it
[ceph: root@host04 /]# ceph osd pool rm default.rgw.meta default.rgw.meta --yes-i-really-really-mean-it
[ceph: root@host04 /]# ceph osd pool rm default.rgw.control default.rgw.control --yes-i-really-really-mean-it
[ceph: root@host04 /]# ceph osd pool rm default.rgw.data.root default.rgw.data.root --yes-i-really-really-mean-it
[ceph: root@host04 /]# ceph osd pool rm default.rgw.gc default.rgw.gc --yes-i-really-really-mean-it
```

e. Ceph 設定データベースを更新します。

構文

```
ceph config set SERVICE_NAME rgw_zone SECONDARY_ZONE_NAME
```

例

```
[ceph: root@host04 /]# ceph config set rgw rgw_zone us-east-2
```

f. 変更をコミットします。

構文

```
radosgw-admin period update --commit
```

例

```
[ceph: root@host04 /]# radosgw-admin period update --commit
```

g. Cephadm シェルの外部で、ストレージクラスターおよびプロセスの FSID を取得します。

例

例

```
[root@host04 ~]#  systemctl list-units | grep ceph
```

h. Ceph Object Gateway デーモンを起動します。

構文

```
systemctl start ceph-FSID@DAEMON_NAME
systemctl enable ceph-FSID@DAEMON_NAME
```

例

```
[root@host04 ~]# systemctl start ceph-62a081a6-88aa-11eb-a367-
001a4a000672@rgw.test_realm.us-east-2.host04.ahdtsw.service
[root@host04 ~]# systemctl enable ceph-62a081a6-88aa-11eb-a367-
001a4a000672@rgw.test_realm.us-east-2.host04.ahdtsw.service
```

3. 必要に応じて、配置仕様を使用して、マルチサイトの Ceph Object Gateway をデプロイします。

構文

```
ceph orch apply rgw NAME --realm=REALM_NAME --zone=PRIMARY_ZONE_NAME --
placement="NUMBER_OF_DAEMONS HOST_NAME_1 HOST_NAME_2"
```

例

```
[ceph: root@host04 /]# ceph orch apply rgw east --realm=test_realm --zone=us-east-1 --
placement="2 host01 host02"
```

検証

- 同期のステータスを確認してデプロイメントを確認します。

例

```
[ceph: root@host04 /]# radosgw-admin sync status
```

10.4. CEPH ORCHESTRATOR を使用した CEPH OBJECT GATEWAY の削除

ceph orch rm コマンドを使用して、Ceph オブジェクトゲートウェイデーモンを削除できます。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- すべてのノードへの root レベルのアクセス。
- ホストがクラスターに追加されている。

- ホストにデプロイされた Ceph オブジェクトゲートウェイデーモンが少なくとも 1 つ含まれます。

手順

1. Cephadm シェルにログインします。

   例

   ```
   [root@host01 ~]# cephadm shell
   ```

2. サービスをリスト表示します。

   例

   ```
   [ceph: root@host01 /]# ceph orch ls
   ```

3. サービスを削除します。

   構文

   ```
   ceph orch rm SERVICE_NAME
   ```

   例

   ```
   [ceph: root@host01 /]# ceph orch rm rgw.test_realm.test_zone_bb
   ```

検証

- ホスト、デーモン、およびプロセスをリスト表示します。

  構文

  ```
  ceph orch ps
  ```

  例

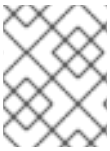
  ```
  [ceph: root@host01 /]# ceph orch ps
  ```

関連情報

- 詳細は、**Red Hat Ceph Storage オペレーションガイド** の **コマンドラインインターフェイスを使用した Ceph オブジェクトゲートウェイのデプロイメント** セクションを参照してください。
- 詳細は、**Red Hat Ceph Storageオペレーションガイド**の **サービス仕様を使用して Ceph オブジェクトゲートウェイのデプロイ** セクションを参照してください。

第11章 CEPH ORCHESTRATOR を使用した NFS-GANESHA ゲートウェイの管理 (利用制限あり)

ストレージ管理者は、バックエンドにて Cephadm と Orchestrator を使用して、NFS-Ganesha ゲートウェイをデプロイできます。Cephadm は、事前定義された RADOS プールおよびオプションの namespace を使用して NFS Ganesha をデプロイします。



注記

この技術には、利用制限があります。詳細は、Deprecated functionality の章を参照してください。

注記

Red Hat は、NFS v4.0+ プロトコルでのみ CephFS エクスポートをサポートします。

本セクションでは、以下の管理タスクを説明します。

- Ceph Orchestrator を使用した NFS-Ganesha クラスターの作成
- コマンドラインインターフェイスを使用した NFS-Ganesha ゲートウェイのデプロイ
- サービス仕様を使用した NFS-Ganesha ゲートウェイのデプロイ
- HA for CephFS/NFS サービスの実装
- Ceph Orchestrator を使用した NFS-Ganesha クラスターの更新
- Ceph Orchestrator を使用した NFS-Ganesha クラスター情報の表示
- Ceph Orchestrator を使用した NFS-Ganesha クラスターログの取得
- Ceph Orchestrator を使用したカスタム NFS-Ganesha の設定
- Ceph Orchestrator を使用したカスタム NFS-Ganesha 設定のリセット
- Ceph Orchestrator を使用した NFS-Ganesha クラスターの削除
- Ceph Orchestrator を使用した NFS Ganesha ゲートウェイの削除

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- すべてのノードへの root レベルのアクセス。
- ホストがクラスターに追加されている。
- すべてのマネージャー、モニター、および OSD デーモンがデプロイされている。

11.1. CEPH ORCHESTRATOR を使用した NFS-GANESHA クラスターの作成

Ceph Orchestrator の **mgr/nfs** モジュールを使用して、NFS-Ganesha クラスターを作成できます。このモジュールは、バックエンドで Cephadm を使用して NFS クラスターをデプロイします。

これにより、すべての NFS-Ganesha デーモンの共通のリカバリープール、**clusterid** に基づく新規ユーザー、および共通の NFS-Ganesha 設定 RADOS オブジェクトが作成されます。

デーモンごとに、新しいユーザーおよび共通の設定がプールに作成されます。すべてのクラスターにはクラスター名に関して異なる namespace がありますが、それらは同じリカバリープールを使用します。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- ホストがクラスターに追加されている。
- すべてのマネージャー、モニター、および OSD デーモンがデプロイされている。

手順

1. Cephadm シェルにログインします。

   例

   ```
   [root@host01 ~]# cephadm shell
   ```

2. **mgr/nfs** モジュールを有効にします。

   例

   ```
   [ceph: root@host01 /]# ceph mgr module enable nfs
   ```

3. クラスターを作成します。

   構文

   ```
   ceph nfs cluster create CLUSTER_NAME ["HOST_NAME_1 HOST_NAME_2 HOST_NAME_3"]
   ```

   CLUSTER_NAME は任意の文字列です。HOST _NAME_1 は、NFS-Ganesha デーモンをデプロイするためにホストを表す任意の文字列です。

   例

   ```
   [ceph: root@host01 /]# ceph nfs cluster create nfsganesha "host01 host02"
   NFS Cluster Created Successful
   ```

   これにより、**host01** および **host02** の 1 つのデーモンと、NFS_Ganesha クラスター **nfsganesha** が作成されます。

検証

- クラスターの詳細をリスト表示します。

例

```
[ceph: root@host01 /]# ceph nfs cluster ls
```

- NFS-Ganesha クラスター情報を表示します。

  構文

  ```
  ceph nfs cluster info CLUSTER_NAME
  ```

  例

  ```
  [ceph: root@host01 /]# ceph nfs cluster info nfsganesha
  ```

関連情報

- 詳細は、**Red Hat Ceph Storage ファイルシステムガイド** の **NFS プロトコルを介した Ceph File System 名前空間のエクスポート** セクションを参照してください。
- 詳細は、**Red Hat Ceph Storage オペレーションガイド** の **サービス仕様を使用した Ceph デーモンのデプロイ** セクションを参照してください。

11.2. コマンドラインインターフェイスを使用した NFS-GANESHA ゲートウェイのデプロイ

バックエンドで Cephadm と Ceph Orchestrator を使用し、配置仕様を使用して NFS-Ganesha ゲートウェイをデプロイできます。この場合、RADOS プールを作成し、namespace を作成してから、ゲートウェイをデプロイする必要があります。

注記

Red Hat は、NFS v4.0+ プロトコルでのみ CephFS エクスポートをサポートします。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- ホストがクラスターに追加されている。
- すべてのマネージャー、モニター、および OSD デーモンがデプロイされている。

手順

1. Cephadm シェルにログインします。

   例

   ```
   [root@host01 ~]# cephadm shell
   ```

2. RADOS プール namespace を作成し、アプリケーションを有効化します。RBD プールの場合、RBD を有効化します。

構文

```
ceph osd pool create POOL_NAME
ceph osd pool application enable POOL_NAME freeform/rgw/rbd/cephfs/nfs
rbd pool init -p POOL_NAME
```

例

```
[ceph: root@host01 /]# ceph osd pool create nfs-ganesha
[ceph: root@host01 /]# ceph osd pool application enable nfs-ganesha nfs
[ceph: root@host01 /]# rbd pool init -p nfs-ganesha
```

3. コマンドラインインターフェイスで配置仕様を使用して NFS-Ganesha ゲートウェイをデプロイします。

構文

```
ceph orch apply nfs SERVICE_ID --placement="NUMBER_OF_DAEMONS HOST_NAME_1 HOST_NAME_2 HOST_NAME_3"
```

例

```
[ceph: root@host01 /]# ceph orch apply nfs foo --placement="2 host01 host02"
```

これにより、**host01** および **host02** の 1 つのデーモンと、NFS-Ganesha クラスター **nfsganesha** がデプロイされます。

検証

- サービスをリスト表示します。

例

```
[ceph: root@host01 /]# ceph orch ls
```

- ホスト、デーモン、およびプロセスをリスト表示します。

構文

```
ceph orch ps --daemon_type=DAEMON_NAME
```

例

```
[ceph: root@host01 /]# ceph orch ps --daemon_type=nfs
```

関連情報

- 詳細は、**Red Hat Ceph Storage オペレーションガイド** の コマンドラインインターフェイスを使用した Ceph デーモンのデプロイ セクションを参照してください。

- 詳細は、**Red Hat Ceph Storage ブロックデバイスガイド** の **ブロックデバイスプールの作成** セクションを参照してください。

11.3. サービス仕様を使用した NFS-GANESHA ゲートウェイのデプロイ

バックエンドで Cephadm と Ceph Orchestrator を使用し、サービス仕様を使用して NFS-Ganesha ゲートウェイをデプロイできます。この場合、RADOS プールを作成し、namespace を作成してから、ゲートウェイをデプロイする必要があります。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- ホストがクラスターに追加されている。

手順

1. **nfs.yaml** ファイルを作成します。

   例

   ```
   [root@host01 ~]# touch nfs.yaml
   ```

2. **nfs.yaml** ファイルを編集し、以下の詳細を含めます。

   構文

   ```
   service_type: nfs
   service_id: SERVICE_ID
   placement:
     hosts:
       - HOST_NAME_1
       - HOST_NAME_2
   ```

   例

   ```
   service_type: nfs
   service_id: foo
   placement:
     hosts:
       - host01
       - host02
   ```

3. YAML ファイルをコンテナー内のディレクトリーにマウントします。

   例

   ```
   [root@host01 ~]# cephadm shell --mount nfs.yaml:/var/lib/ceph/nfs.yaml
   ```

4. RADOS プールと namespace を作成し、RBD を有効にします。

   構文

```
ceph osd pool create POOL_NAME
ceph osd pool application enable POOL_NAME rbd
rbd pool init -p POOL_NAME
```

例

```
[ceph: root@host01 /]# ceph osd pool create nfs-ganesha
[ceph: root@host01 /]# ceph osd pool application enable nfs-ganesha rbd
[ceph: root@host01 /]# rbd pool init -p nfs-ganesha
```

5. そのディレクトリーに移動します。

   例

   ```
   [ceph: root@host01 /]# cd /var/lib/ceph/
   ```

6. サービス仕様を使用して NFS-Ganesha ゲートウェイをデプロイします。

   構文

   ```
   ceph orch apply -i FILE_NAME.yaml
   ```

   例

   ```
   [ceph: root@host01 ceph]# ceph orch apply -i nfs.yaml
   ```

検証

- サービスをリスト表示します。

  例

  ```
  [ceph: root@host01 /]# ceph orch ls
  ```

- ホスト、デーモン、およびプロセスをリスト表示します。

  構文

  ```
  ceph orch ps --daemon_type=DAEMON_NAME
  ```

  例

  ```
  [ceph: root@host01 /]# ceph orch ps --daemon_type=nfs
  ```

関連情報

- 詳細は、**Red Hat Ceph Storage ブロックデバイスガイド** の **ブロックデバイスプールの作成** セクションを参照してください。

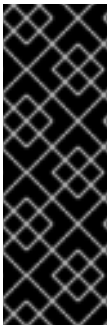
11.4. HA FOR CEPHFS/NFS サービスの実装 (テクノロジープレビュー)

--ingress フラグを使用し、仮想 IP アドレスを指定することで、高可用性 (HA) フロントエンド、仮想 IP、およびロードバランサーを備えた NFS をデプロイできます。これにより、**keepalived** と **haproxy** の組み合わせがデプロイされ、NFS サービスに高可用性の NFS フロントエンドが提供されます。

--ingress フラグを指定してクラスターを作成すると、NFS サーバーの負荷分散と高可用性を提供するために、Ingress サービスが追加でデプロイされます。仮想 IP は、すべての NFS クライアントがマウントに使用できる既知の安定した NFS エンドポイントを提供するために使用されます。Ceph は、仮想 IP 上の NFS トラフィックを適切なバックエンド NFS サーバーにリダイレクトする詳細を処理し、失敗した場合に NFS サーバーを再デプロイします。

既存のサービスに Ingress サービスをデプロイすると、以下が提供されます。

- NFS サーバーへのアクセスに使用できる安定した仮想 IP。
- 複数の NFS ゲートウェイに負荷を分散します。
- ホストに障害が発生した場合のホスト間のフェイルオーバー。

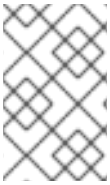


重要

HA for CephFS/NFS はテクノロジープレビュー機能のみになります。テクノロジープレビュー機能は、実稼働環境での Red Hat サービスレベルアグリーメント (SLA) ではサポートされておらず、機能的に完全ではない可能性があるため、Red Hat では実稼働環境での使用を推奨していません。テクノロジープレビューの機能は、最新の製品機能をいち早く提供して、開発段階で機能のテストを行いフィードバックを提供していただくことを目的としています。詳細は、Red Hat 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。

重要

ingress サービスが NFS クラスターの前にデプロイされると、バックエンドの NFS-ganesha サーバーは、クライアントの IP アドレスではなく、haproxy の IP アドレスを認識します。その結果、IP アドレスに基づいてクライアントアクセスを制限している場合、NFS エクスポートのアクセス制限は想定どおりに機能しません。



注記

クライアントにサービスを提供しているアクティブな NFS サーバーがダウンした場合、クライアントの I/O は、アクティブな NFS サーバーの代替サーバーがオンラインになり、NFS クラスターが再びアクティブになるまで中断されます。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- ホストがクラスターに追加されている。
- すべてのマネージャー、モニター、および OSD デーモンがデプロイされている。
- NFS モジュールが有効になっていることを確認している。

手順

1. Cephadm シェルにログインします。

例

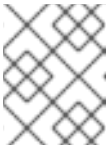
```
[root@host01 ~]# cephadm shell
```

2. **--ingress** フラグを使用して NFS クラスターを作成します。

構文

```
ceph nfs cluster create CLUSTER_ID [PLACEMENT] [--port PORT_NUMBER] [--ingress --virtual-ip IP_ADDRESS/CIDR_PREFIX]
```

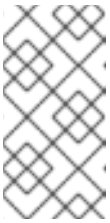
- **CLUSTER_ID** を一意の文字列に置き換えて、NFS Ganesha クラスターに名前を付けます。
- **PLACEMENT** を、デプロイする NFS サーバーの数と、NFS Ganesha デーモンコンテナーをデプロイするホスト (1 つまたは複数) に置き換えます。
- **--port PORT_NUMBER** フラグを使用して、デフォルトのポート 12049 以外のポートに NFS をデプロイします。



注記

ingress モードでは、高可用性プロキシーはポート 2049 を使用し、NFS サービスは 12049 にデプロイされます。

- **--ingress** フラグを **--virtual-ip** フラグと組み合わせると、高可用性フロントエンド (仮想 IP およびロードバランサー) を使用して NFS がデプロイされます。
- **--virtual-ip IP_ADDRESS** を IP アドレスに置き換えて、すべてのクライアントが NFS エクスポートをマウントするために使用できる既知の安定した NFS エンドポイントを提供します。**--virtual-ip** には、CIDR 接頭辞長を含める必要があります。仮想 IP は通常、同じサブネット内に既存の IP を持つ最初に識別されたネットワークインターフェイスで設定されます。

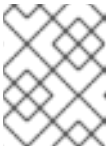


注記

NFS サービスに割り当てるホストの数は、デプロイを要求するアクティブな NFS サーバーの数 (**placement: count** パラメーターで指定) よりも大きくする必要があります。以下の例では、1 つのアクティブな NFS サーバーが要求され、2 つのホストが割り当てられます。

例

```
[ceph: root@host01 /]# ceph nfs cluster create mycephnfs "1 host02 host03" --ingress --virtual-ip 10.10.128.75/22
```



注記

NFS デーモンと Ingress サービスのデプロイメントは非同期であり、サービスが完全に開始される前にコマンドが返される場合があります。

3. サービスが正常に開始されたことを確認します。

構文

```
ceph orch ls --service_name=nfs.CLUSTER_ID
ceph orch ls --service_name=ingress.nfs.CLUSTER_ID
```

例

```
[ceph: root@host01 /]# ceph orch ls --service_name=nfs.mycephnfs

NAME          PORTS    RUNNING  REFRESHED  AGE  PLACEMENT
nfs.mycephnfs  ?:12049      2/2  0s ago     20s  host02;host03

[ceph: root@host01 /]# ceph orch ls --service_name=ingress.nfs.mycephnfs

NAME                 PORTS                 RUNNING  REFRESHED  AGE  PLACEMENT
ingress.nfs.mycephnfs  10.10.128.75:2049,9049      4/4  46s ago    73s  count:2
```

検証

- IP エンドポイント、個々の NFS デーモンの IP、および **ingress** サービスの仮想 IP を表示します。

構文

```
ceph nfs cluster info CLUSTER_NAME
```

例

```
[ceph: root@host01 /]# ceph nfs cluster info mycephnfs

{
    "mycephnfs": {
        "virtual_ip": "10.10.128.75",
        "backend": [
            {
                "hostname": "host02",
                "ip": "10.10.128.69",
                "port": 12049
            },
            {
                "hostname": "host03",
                "ip": "10.10.128.70",
                "port": 12049
            }
        ],
        "port": 2049,
        "monitor_port": 9049
    }
}
```

- ホストおよびプロセスをリスト表示します。

例

```
[ceph: root@host01 /]# ceph orch ps | grep nfs

haproxy.nfs.cephnfs.host01.rftylv     host01  *:2049,9000  running (11m)    10m ago  11m
23.2M        -  2.2.19-7ea3822   5e6a41d77b38  f8cc61dc827e
haproxy.nfs.cephnfs.host02.zhtded     host02  *:2049,9000  running (11m)    53s ago  11m
21.3M        -  2.2.19-7ea3822   5e6a41d77b38  4cad324e0e23
keepalived.nfs.cephnfs.host01.zktmsk  host01               running (11m)    10m ago  11m
2349k        -  2.1.5           18fa163ab18f  66bf39784993
keepalived.nfs.cephnfs.host02.vyycvp  host02               running (11m)    53s ago  11m
2349k        -  2.1.5           18fa163ab18f  1ecc95a568b4
nfs.cephnfs.0.0.host02.fescmw         host02  *:12049      running (14m)     3m ago  14m
76.9M        -  3.5             cef6e7959b0a  bb0e4ee9484e
nfs.cephnfs.1.0.host03.avaddf         host03  *:12049      running (14m)     3m ago  14m
74.3M        -  3.5             cef6e7959b0a  ea02c0c50749
```

関連情報

- クライアントホストに NFS エクスポートをマウントする方法の詳細は、**Red Hat Ceph Storage ファイルシステムガイド** の NFS プロトコルを介した Ceph File System 名前空間のエクスポート セクションを参照してください。

11.5. スタンドアロン CEPHFS/NFS CLUSTER FOR HA のアップグレード

ストレージ管理者は、既存の NFS サービスに **ingress** サービスをデプロイすることで、スタンドアロンストレージクラスターを高可用性 (HA) クラスターにアップグレードできます。

前提条件

- 既存の NFS サービスを備えた実行中の Red Hat Ceph Storage クラスター。
- ホストがクラスターに追加されている。
- すべてのマネージャー、モニター、および OSD デーモンがデプロイされている。
- NFS モジュールが有効になっていることを確認している。

手順

1. Cephadm シェルにログインします。

   例

   ```
   [root@host01 ~]# cephadm shell
   ```

2. 既存の NFS クラスターをリスト表示します。

   例

   ```
   [ceph: root@host01 /]# ceph nfs cluster ls

   mynfs
   ```

注記

スタンドアロン NFS クラスターが 1 つのノードに作成されている場合、HA 用に 2 つ以上のノードに増やす必要があります。NFS サービスを増やすには、**nfs.yaml** ファイルを編集し、同じポート番号で配置を増やします。

NFS サービスに割り当てるホストの数は、デプロイを要求するアクティブな NFS サーバーの数 (**placement: count** パラメーターで指定) よりも大きくする必要があります。

構文

```
service_type: nfs
service_id: SERVICE_ID
placement:
  hosts:
    - HOST_NAME_1
    - HOST_NAME_2
  count: COUNT
spec:
  port: PORT_NUMBER
```

例

```
service_type: nfs
service_id: mynfs
placement:
  hosts:
    - host02
    - host03
  count: 1
spec:
  port: 12345
```

この例では、既存の NFS サービスがポート **12345** で実行されており、同じポートを使用して追加のノードが NFS クラスターに追加されます。

nfs.yaml サービス仕様の変更を適用して、2 ノード NFS サービスにアップグレードします。

例

```
[ceph: root@host01 ceph]# ceph orch apply -i nfs.yaml
```

3. 既存の NFS クラスター ID を使用して **ingress.yaml** 仕様ファイルを編集します。

構文

```
service_type: SERVICE_TYPE
service_id: SERVICE_ID
placement:
  count: PLACEMENT
spec:
  backend_service: SERVICE_ID_BACKEND (1)
```

```
frontend_port: FRONTEND_PORT
monitor_port: MONITOR_PORT 2
virtual_ip: VIRTUAL_IP_WITH_CIDR
```

1 **SERVICE_ID_BACKEND** には、同じホストに配置される可能性がある Ingress サービスとの競合を避けるために、**2049** 以外の **PORT** プロパティーを含める必要があります。

2 **MONITOR_PORT** は、haproxy ロードステータスページにアクセスするために使用されます。

例

```
service_type: ingress
service_id: nfs.mynfs
placement:
  count: 2
spec:
  backend_service: nfs.mynfs
  frontend_port: 2049
  monitor_port: 9000
  virtual_ip: 10.10.128.75/22
```

4. Ingress サービスをデプロイします。

例

```
[ceph: root@host01 /]# ceph orch apply -i ingress.yaml
```

注記

NFS デーモンと Ingress サービスのデプロイメントは非同期であり、サービスが完全に開始される前にコマンドが返される場合があります。

5. Ingress サービスが正常に開始されたことを確認します。

構文

```
ceph orch ls --service_name=ingress.nfs.CLUSTER_ID
```

例

```
[ceph: root@host01 /]# ceph orch ls --service_name=ingress.nfs.mynfs

NAME                 PORTS                 RUNNING  REFRESHED  AGE  PLACEMENT
ingress.nfs.mynfs  10.10.128.75:2049,9000      4/4   4m ago     22m  count:2
```

検証

- IP エンドポイント、個々の NFS デーモンの IP、および **ingress** サービスの仮想 IP を表示します。

構文

```
ceph nfs cluster info CLUSTER_NAME
```

例

```
[ceph: root@host01 /]# ceph nfs cluster info mynfs

{
    "mynfs": {
        "virtual_ip": "10.10.128.75",
        "backend": [
            {
                "hostname": "host02",
                "ip": "10.10.128.69",
                "port": 12049
            },
            {
                "hostname": "host03",
                "ip": "10.10.128.70",
                "port": 12049
            }
        ],
        "port": 2049,
        "monitor_port": 9049
    }
}
```

- ホストおよびプロセスをリスト表示します。

例

```
[ceph: root@host01 /]# ceph orch ps | grep nfs

haproxy.nfs.mynfs.host01.ruyyhq     host01  *:2049,9000  running (27m)     6m ago  34m
9.85M        -  2.2.19-7ea3822   5e6a41d77b38  328d27b3f706
haproxy.nfs.mynfs.host02.ctrhha     host02  *:2049,9000  running (34m)     6m ago  34m
4944k        -  2.2.19-7ea3822   5e6a41d77b38  4f4440dbfde9
keepalived.nfs.mynfs.host01.fqgjxd  host01               running (27m)     6m ago  34m    31.2M
-  2.1.5            18fa163ab18f  0e22b2b101df
keepalived.nfs.mynfs.host02.fqzkxb  host02               running (34m)     6m ago  34m    17.5M
-  2.1.5            18fa163ab18f  c1e3cc074cf8
nfs.mynfs.0.0.host02.emoaut         host02  *:12345      running (37m)     6m ago  37m    82.7M
-  3.5              91322de4f795  2d00faaa2ae5
nfs.mynfs.1.0.host03.nsxcfd         host03  *:12345      running (37m)     6m ago  37m    81.1M
-  3.5              91322de4f795  d4bda4074f17
```

関連情報

- クライアントホストに NFS エクスポートをマウントする方法の詳細は、**Red Hat Ceph Storage ファイルシステムガイド** の NFS プロトコルを介した Ceph File System 名前空間のエクスポート セクションを参照してください。

11.6. 仕様ファイルを使用した HA FOR CEPHFS/NFS のデプロイ

仕様ファイルを使用して HA for CephFS/NFS をデプロイするには、最初に NFS サービスをデプロイしてから、同じ NFS サービスに **ingress** をデプロイします。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- ホストがクラスターに追加されている。
- すべてのマネージャー、モニター、および OSD デーモンがデプロイされている。
- NFS モジュールが有効になっていることを確認している。

手順

1. Cephadm シェルにログインします。

   例

   ```
   [root@host01 ~]# cephadm shell
   ```

2. NFS モジュールが有効になっていることを確認します。

   例

   ```
   [ceph: root@host01 /]# ceph mgr module ls | more
   ```

3. Cephadm シェルを終了し、**nfs.yaml** ファイルを作成します。

   例

   ```
   [root@host01 ~]# touch nfs.yaml
   ```

4. **nfs.yaml** ファイルを編集し、以下の詳細を含めます。

   構文

   ```
   service_type: nfs
   service_id: SERVICE_ID
   placement:
     hosts:
       - HOST_NAME_1
       - HOST_NAME_2
     count: COUNT
   spec:
     port: PORT_NUMBER
   ```

注記

NFS サービスに割り当てるホストの数は、デプロイを要求するアクティブな NFS サーバーの数 (**placement: count** パラメーターで指定) よりも大きくする必要があります。

例

```
service_type: nfs
service_id: cephfsnfs
placement:
  hosts:
    - host02
    - host03
  count: 1
spec:
  port: 12345
```

この例では、サーバーは、host02 と host03 のデフォルトポート **2049** ではなく、デフォルト以外のポート **12345** で実行されます。

5. YAML ファイルをコンテナー内のディレクトリーにマウントします。

例

```
[root@host01 ~]# cephadm shell --mount nfs.yaml:/var/lib/ceph/nfs.yaml
```

6. Cephadm シェルにログインし、ディレクトリーに移動します。

例

```
[ceph: root@host01 /]# cd /var/lib/ceph/
```

7. サービス仕様を使用して NFS サービスをデプロイします。

構文

```
ceph orch apply -i FILE_NAME.yaml
```

例

```
[ceph: root@host01 ceph]# ceph orch apply -i nfs.yaml
```

注記

NFS サービスのデプロイメントは非同期で行われるため、サービスが完全に開始される前にコマンドが返ってくる可能性があります。

8. NFS サービスが正常に開始されたことを確認します。

構文

```
ceph orch ls --service_name=nfs.CLUSTER_ID
```

例

```
[ceph: root@host01 /]# ceph orch ls --service_name=nfs.cephfsnfs

NAME          PORTS    RUNNING  REFRESHED  AGE  PLACEMENT
nfs.cephfsnfs  ?:12345      2/2  3m ago     13m  host02;host03
```

9. Cephadm シェルを終了し、**ingress.yaml** ファイルを作成します。

例

```
[root@host01 ~]# touch ingress.yaml
```

10. **ingress.yaml** ファイルを編集して、以下の詳細を含めます。

構文

```
service_type: SERVICE_TYPE
service_id: SERVICE_ID
placement:
  count: PLACEMENT
spec:
  backend_service: SERVICE_ID_BACKEND
  frontend_port: FRONTEND_PORT
  monitor_port: MONITOR_PORT
  virtual_ip: VIRTUAL_IP_WITH_CIDR
```

例

```
service_type: ingress
service_id: nfs.cephfsnfs
placement:
  count: 2
spec:
  backend_service: nfs.cephfsnfs
  frontend_port: 2049
  monitor_port: 9000
  virtual_ip: 10.10.128.75/22
```

注記

この例では、**placement: count: 2** は **keepalived** および **haproxy** サービスをランダムなノードにデプロイします。**keepalived** と **haproxy** をデプロイするノードを指定するには、**placement: hosts** オプションを使用します。

例

```
service_type: ingress
service_id: nfs.cephfsnfs
placement:
  hosts:
    - host02
    - host03
```

11. YAML ファイルをコンテナー内のディレクトリーにマウントします。

    例

    ```
    [root@host01 ~]# cephadm shell --mount ingress.yaml:/var/lib/ceph/ingress.yaml
    ```

12. Cephadm シェルにログインし、ディレクトリーに移動します。

    例

    ```
    [ceph: root@host01 /]# cd /var/lib/ceph/
    ```

13. サービス仕様を使用して **ingress** サービスをデプロイします。

    構文

    ```
    ceph orch apply -i FILE_NAME.yaml
    ```

    例

    ```
    [ceph: root@host01 ceph]# ceph orch apply -i ingress.yaml
    ```

14. Ingress サービスが正常に開始されたことを確認します。

    構文

    ```
    ceph orch ls --service_name=ingress.nfs.CLUSTER_ID
    ```

    例

    ```
    [ceph: root@host01 /]# ceph orch ls --service_name=ingress.nfs.cephfsnfs

    NAME                 PORTS                 RUNNING  REFRESHED  AGE  PLACEMENT
    ingress.nfs.cephfsnfs  10.10.128.75:2049,9000      4/4   4m ago     22m  count:2
    ```

検証

- IP エンドポイント、個々の NFS デーモンの IP、および **ingress** サービスの仮想 IP を表示します。

  構文

  ```
  ceph nfs cluster info CLUSTER_NAME
  ```

  例

  ```
  [ceph: root@host01 /]# ceph nfs cluster info cephfsnfs

  {
     "cephfsnfs": {
        "virtual_ip": "10.10.128.75",
        "backend": [
           {
              "hostname": "host02",
              "ip": "10.10.128.69",
              "port": 12345
           },
           {
              "hostname": "host03",
              "ip": "10.10.128.70",
              "port": 12345
           }
        ],
        "port": 2049,
        "monitor_port": 9049
     }
  }
  ```

- ホストおよびプロセスをリスト表示します。

  例

  ```
  [ceph: root@host01 /]# ceph orch ps | grep nfs

  haproxy.nfs.cephfsnfs.host01.ruyyhq     host01  *:2049,9000  running (27m)     6m ago  34m
  9.85M        -  2.2.19-7ea3822   5e6a41d77b38  328d27b3f706
  haproxy.nfs.cephfsnfs.host02.ctrhha     host02  *:2049,9000  running (34m)     6m ago  34m
  4944k        -  2.2.19-7ea3822   5e6a41d77b38  4f4440dbfde9
  keepalived.nfs.cephfsnfs.host01.fqgjxd  host01               running (27m)     6m ago  34m
  31.2M        -  2.1.5            18fa163ab18f  0e22b2b101df
  keepalived.nfs.cephfsnfs.host02.fqzkxb  host02               running (34m)     6m ago  34m
  17.5M        -  2.1.5            18fa163ab18f  c1e3cc074cf8
  nfs.cephfsnfs.0.0.host02.emoaut         host02  *:12345      running (37m)     6m ago  37m
  82.7M        -  3.5              91322de4f795  2d00faaa2ae5
  nfs.cephfsnfs.1.0.host03.nsxcfd         host03  *:12345      running (37m)     6m ago  37m
  81.1M        -  3.5              91322de4f795  d4bda4074f17
  ```

関連情報

- クライアントホストに NFS エクスポートをマウントする方法の詳細は、**Red Hat Ceph Storage ファイルシステムガイド** の NFS プロトコルを介した Ceph File System 名前空間のエクスポート セクションを参照してください。

11.7. CEPH ORCHESTRATOR を使用した NFS-GANESHA クラスターの更新

バックエンドで Ceph Orchestrator と Cephadm を使用して、ホスト上のデーモンの配置を変更して、NFS-Ganesha クラスターを更新できます。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- ホストがクラスターに追加されている。
- すべてのマネージャー、モニター、および OSD デーモンがデプロイされている。
- **mgr/nfs** モジュールを使用して作成された NFS-Ganesha クラスター。

手順

1. Cephadm シェルにログインします。

   例

   ```
   [root@host01 ~]# cephadm shell
   ```

2. クラスターを更新します。

   構文

   ```
   ceph orch apply nfs CLUSTER_NAME
   ["HOST_NAME_1,HOST_NAME_2,HOST_NAME_3"]
   ```

   CLUSTER_NAME は任意の文字列です。**HOST _NAME_1** は、デプロイされた NFS-Ganesha デーモンをするためにホストを表す任意の文字列です。

   例

   ```
   [ceph: root@host01 /]# ceph orch apply nfs nfsganesha "host02"
   ```

   これにより、**host02** 上の **nfsganesha** クラスターが更新されます。

検証

- クラスターの詳細をリスト表示します。

  例

  ```
  [ceph: root@host01 /]# ceph nfs cluster ls
  ```

- NFS-Ganesha クラスター情報を表示します。

構文

```
ceph nfs cluster info CLUSTER_NAME
```

例

```
[ceph: root@host01 /]# ceph nfs cluster info nfsganesha
```

- ホスト、デーモン、およびプロセスをリスト表示します。

構文

```
ceph orch ps --daemon_type=DAEMON_NAME
```

例

```
[ceph: root@host01 /]# ceph orch ps --daemon_type=nfs
```

関連情報

- 詳細は、**Red Hat Ceph Storage オペレーションガイド** の Ceph Orchestrator を使用した NFS-Ganesha クラスターの作成 セクションを参照してください。

11.8. CEPH ORCHESTRATOR を使用した NFS-GANESHA クラスター情報の確認

Ceph Orchestrator を使用して、NFS-Ganesha クラスターの情報を確認できます。すべての NFS Ganesha クラスターや、ポート、IP アドレス、およびクラスターが作成されたホストの名前のある特定のクラスターに関する情報を取得できます。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- ホストがクラスターに追加されている。
- すべてのマネージャー、モニター、および OSD デーモンがデプロイされている。
- **mgr/nfs** モジュールを使用して作成された NFS-Ganesha クラスター。

手順

1. Cephadm シェルにログインします。

   例

   ```
   [root@host01 ~]# cephadm shell
   ```

2. NFS-Ganesha クラスター情報を表示します。

   構文

   -

```
ceph nfs cluster info CLUSTER_NAME
```

例

```
[ceph: root@host01 /]# ceph nfs cluster info nfsganesha

{
    "nfsganesha": [
        {
            "hostname": "host02",
            "ip": [
                "10.10.128.70"
            ],
            "port": 2049
        }
    ]
}
```

関連情報

- 詳細は、**Red Hat Ceph Storage オペレーションガイド** の Ceph Orchestrator を使用した NFS-Ganesha クラスターの作成 セクションを参照してください。

11.9. CEPH ORCHESTRATOR を使用した NFS-GANESHA CLUTSER ログの取得

Ceph Orchestrator を使用すると、NFS-Ganesha クラスターログを取得できます。サービスがデプロイされているノードにいる必要があります。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- NFS がデプロイされたノードに Cephadm がインストールされている。
- すべてのノードへの root レベルのアクセス。
- ホストがクラスターに追加されている。
- **mgr/nfs** モジュールを使用して作成された NFS-Ganesha クラスター。

手順

1. root ユーザーとして、ストレージクラスターの **FSID** を取得します。

   例

   ```
   [root@host03 ~]# cephadm ls
   ```

   FSID とサービスの名前をコピーします。

2. ログを取得します。

構文

```
cephadm logs --fsid FSID --name SERVICE_NAME
```

例

```
[root@host03 ~]# cephadm logs --fsid 499829b4-832f-11eb-8d6d-001a4a000635 --name
nfs.foo.host03
```

関連情報

- 詳細は、**Red Hat Ceph Storage オペレーションガイド** の **配置仕様を使用した Ceph デーモンのデプロイメント** セクションを参照してください。

11.10. CEPH ORCHESTRATOR を使用したカスタム NFS-GANESHA 設定の設定

NFS-Ganesha クラスターはデフォルトの設定ブロックで定義されます。Ceph Orchestrator を使用すると、設定をカスタマイズでき、デフォルトの設定ブロックよりも優先されます。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- ホストがクラスターに追加されている。
- すべてのマネージャー、モニター、および OSD デーモンがデプロイされている。
- **mgr/nfs** モジュールを使用して作成された NFS-Ganesha クラスター。

手順

1. Cephadm シェルにログインします。

   例

   ```
   [root@host01 ~]# cephadm shell
   ```

2. 以下は、NFS-Ganesha クラスターのデフォルト設定の例です。

   例

   ```
   # {{ cephadm_managed }}
   NFS_CORE_PARAM {
           Enable_NLM = false;
           Enable_RQUOTA = false;
           Protocols = 4;
   }

   MDCACHE {
           Dir_Chunk = 0;
   }
   ```

```
EXPORT_DEFAULTS {
        Attr_Expiration_Time = 0;
}

NFSv4 {
        Delegations = false;
        RecoveryBackend = 'rados_cluster';
        Minor_Versions = 1, 2;
}

RADOS_KV {
        UserId = "{{ user }}";
        nodeid = "{{ nodeid}}";
        pool = "{{ pool }}";
        namespace = "{{ namespace }}";
}

RADOS_URLS {
        UserId = "{{ user }}";
        watch_url = "{{ url }}";
}

RGW {
        cluster = "ceph";
        name = "client.{{ rgw_user }}";
}

%url    {{ url }}
```

3. NFS-Ganesha クラスター設定をカスタマイズします。以下は、設定をカスタマイズする 2 つの例です。

   - ログレベルを変更します。

     例

     ```
     LOG {
      COMPONENTS {
          ALL = FULL_DEBUG;
      }
     }
     ```

   - カスタムエクスポートブロックを追加します。

     a. ユーザーを作成します。

     注記

     FSAL ブロックで指定されたユーザーは、NFS-Ganesha デーモンが Ceph ク ラスターにアクセスするための適切な上限を設定する必要があります。

     構文

```
ceph auth get-or-create client.USER_ID mon 'allow r' osd 'allow rw pool=.nfs
namespace=NFS_CLUSTER_NAME, allow rw tag cephfs data=FS_NAME' mds
'allow rw path=EXPORT_PATH'
```

例

```
[ceph: root@host01 /]# ceph auth get-or-create client.f64f341c-655d-11eb-8778-
fa163e914bcc mon 'allow r' osd 'allow rw pool=.nfs namespace=nfs_cluster_name,
allow rw tag cephfs data=fs_name' mds 'allow rw path=export_path'
```

b. 以下のディレクトリーに移動します。

構文

```
cd /var/lib/ceph/DAEMON_PATH/
```

例

```
[ceph: root@host01 /]# cd /var/lib/ceph/nfs/
```

nfs ディレクトリーが存在しない場合は、パス上に作成します。

c. 新しい設定ファイルを作成します。

構文

```
touch PATH_TO_CONFIG_FILE
```

例

```
[ceph: root@host01 nfs]#  touch nfs-ganesha.conf
```

d. カスタムエクスポートブロックを追加して設定ファイルを編集します。1つのエクスポートを作成し、これは Ceph NFS エクスポートインターフェイスによって管理されます。

構文

```
EXPORT {
  Export_Id = NUMERICAL_ID;
  Transports = TCP;
  Path = PATH_WITHIN_CEPHFS;
  Pseudo = BINDING;
  Protocols = 4;
  Access_Type = PERMISSIONS;
  Attr_Expiration_Time = 0;
  Squash = None;
  FSAL {
    Name = CEPH;
    Filesystem = "FILE_SYSTEM_NAME";
    User_Id = "USER_NAME";
```

```
    Secret_Access_Key = "USER_SECRET_KEY";
  }
}
```

例

```
EXPORT {
  Export_Id = 100;
  Transports = TCP;
  Path = /;
  Pseudo = /ceph/;
  Protocols = 4;
  Access_Type = RW;
  Attr_Expiration_Time = 0;
  Squash = None;
  FSAL {
    Name = CEPH;
    Filesystem = "filesystem name";
    User_Id = "user id";
    Secret_Access_Key = "secret key";
  }
}
```

4. 新しい設定をクラスターに適用します。

   構文

   ```
   ceph nfs cluster config set CLUSTER_NAME -i PATH_TO_CONFIG_FILE
   ```

   例

   ```
   [ceph: root@host01 nfs]# ceph nfs cluster config set nfs-ganesha -i /var/lib/ceph/nfs/nfs-ganesha.conf
   ```

   これにより、カスタム設定のサービスも再起動されます。

検証

- サービスをリスト表示します。

  例

  ```
  [ceph: root@host01 /]# ceph orch ls
  ```

- ホスト、デーモン、およびプロセスをリスト表示します。

  構文

  ```
  ceph orch ps --daemon_type=DAEMON_NAME
  ```

  例

```
[ceph: root@host01 /]# ceph orch ps --daemon_type=nfs
```

- カスタム設定を確認します。

  構文

  ```
  /bin/rados -p POOL_NAME -N CLUSTER_NAME get userconf-nfs.CLUSTER_NAME -
  ```

  例

  ```
  [ceph: root@host01 /]# /bin/rados -p nfs-ganesha -N nfsganesha get userconf-
  nfs.nfsganesha -
  ```

関連情報

- 詳細は、**Red Hat Ceph Storage オペレーションガイド** の Resetting custom NFS-Ganesha configuration using the Ceph Orchestrator セクションを参照してください。

11.11. CEPH ORCHESTRATOR を使用したカスタム NFS-GANESHA 設定のリセット

Ceph Orchestrator を使用すると、ユーザー定義の設定をデフォルト設定にリセットできます。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- ホストがクラスターに追加されている。
- すべてのマネージャー、モニター、および OSD デーモンがデプロイされている。
- **mgr/nfs** モジュールを使用してデプロイされた NFS-Ganesha
- カスタム NFS クラスターの設定がセットアップされている。

手順

1. Cephadm シェルにログインします。

   例

   ```
   [root@host01 ~]# cephadm shell
   ```

2. NFS_Ganesha 設定をリセットします。

   構文

   ```
   ceph nfs cluster config reset CLUSTER_NAME
   ```

   例

```
[ceph: root@host01 /]# ceph nfs cluster config reset nfs-cephfs
```

検証

- サービスをリスト表示します。

  例

  ```
  [ceph: root@host01 /]# ceph orch ls
  ```

- ホスト、デーモン、およびプロセスをリスト表示します。

  構文

  ```
  ceph orch ps --daemon_type=DAEMON_NAME
  ```

  例

  ```
  [ceph: root@host01 /]# ceph orch ps --daemon_type=nfs
  ```

- カスタム設定が削除されていることを確認します。

  構文

  ```
  /bin/rados -p POOL_NAME -N CLUSTER_NAME get userconf-nfs.CLUSTER_NAME -
  ```

  例

  ```
  [ceph: root@host01 /]# /bin/rados -p nfs-ganesha -N nfsganesha get userconf-
  nfs.nfsganesha -
  ```

関連情報

- 詳細は、**Red Hat Ceph Storage オペレーションガイド** の Ceph Orchestrator を使用した NFS-Ganesha クラスターの作成 セクションを参照してください。
- Operation ガイド詳細は、**Red Hat Ceph Storage オペレーションガイド** の Setting custom NFS-Ganesha configuration using the Ceph Orchestrator セクションを参照してください。

11.12. CEPH ORCHESTRATOR を使用した NFS-GANESHA クラスターの削除

バックエンドで Cephadm と Ceph Orchestrator を使用すると、NFS-Ganesha クラスターを削除できます。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- ホストがクラスターに追加されている。

- すべてのマネージャー、モニター、および OSD デーモンがデプロイされている。
- **mgr/nfs** モジュールを使用して作成された NFS-Ganesha クラスター。

手順

1. Cephadm シェルにログインします。

   例

   ```
   [root@host01 ~]# cephadm shell
   ```

2. クラスターを削除します。

   構文

   ```
   ceph nfs cluster rm CLUSTER_NAME
   ```

   CLUSTER_NAME は任意の文字列です。

   例

   ```
   [ceph: root@host01 /]# ceph nfs cluster rm nfsganesha
   NFS Cluster Deleted Successfully
   ```

注記

delete オプションは非推奨になっており、NFS クラスターを削除するには **rm** を使用する必要があります。

検証

- クラスターの詳細をリスト表示します。

  例

  ```
  [ceph: root@host01 /]# ceph nfs cluster ls
  ```

関連情報

- 詳細は、**Red Hat Ceph Storage File System Guide** の Exporting Ceph File System namespaces over the NFS protocol セクションを参照してください。
- 詳細は、**Red Hat Ceph Storage オペレーションガイド** の Ceph Orchestrator を使用した NFS-Ganesha クラスターの作成 セクションを参照してください。

11.13. CEPH ORCHESTRATOR を使用した NFS-GANESHA ゲートウェイの削除

ceph orch rm コマンドを使用して、NFS-Ganesha ゲートウェイを削除できます。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- すべてのノードへの root レベルのアクセス。
- ホストがクラスターに追加されている。
- ホストにデプロイされた 1 つ以上の NFS-Ganesha ゲートウェイ。

手順

1. Cephadm シェルにログインします。

   例

   ```
   [root@host01 ~]# cephadm shell
   ```

2. サービスをリスト表示します。

   例

   ```
   [ceph: root@host01 /]# ceph orch ls
   ```

3. サービスを削除します。

   構文

   ```
   ceph orch rm SERVICE_NAME
   ```

   例

   ```
   [ceph: root@host01 /]# ceph orch rm nfs.foo
   ```

検証

- ホスト、デーモン、およびプロセスをリスト表示します。

  構文

  ```
  ceph orch ps
  ```

  例

  ```
  [ceph: root@host01 /]# ceph orch ps
  ```

関連情報

- 詳細は、**Red Hat Ceph Storage オペレーションガイド** の **サービス仕様を使用した Ceph デーモンのデプロイ** セクションを参照してください。

- 詳細は、**Red Hat Ceph Storage オペレーションガイド** の **サービス仕様を使用した NFS-Ganesha ゲートウェイのデプロイ** セクションを参照してください。

11.14. KERBEROS インテグレーション

Kerberos は、信頼されないネットワーク全体でユーザーをサーバーに認証したり、サーバーをユーザーに認証したりする集中認証サーバーを提供するコンピューターネットワークセキュリティープロトコルです。Kerberos 認証では、サーバーとデータベースがクライアント認証に使用されます。

11.14.1. KDC のセットアップ (要件に応じて)

Kerberos は、キー配布センター (KDC) として知られるサードパーティーの信頼されたサーバーとして実行され、ネットワーク上の各ユーザーとサービスがプリンシパルとなります。KDC は、すべてのクライアント (ユーザープリンシパル、サービスプリンシパル) に関する情報を保持します。したがって、セキュアである必要があります。Kerberos セットアップでは、KDC が単一障害点となるため、1 つのマスター KDC と複数のスレーブ KDC を使用することを推奨します。

前提条件

/etc/hosts ファイルに次の変更が加えられているかどうかを確認します。必要に応じてドメイン名を追加します。

```
[root@chost ~]# cat /etc/hosts

127.0.0.1   localhost localhost.localdomain localhost4 localhost4.localdomain4
::1         localhost localhost.localdomain localhost6 localhost6.localdomain6
10.0.208.97         ceph-node1-installer.ibm.com ceph-node1-installer
10.0.210.243 ceph-node2.ibm.com ceph-node2
10.0.208.63         ceph-node3.ibm.com ceph-node3
10.0.210.222 ceph-node4.ibm.com ceph-node4
10.0.210.235 ceph-node5.ibm.com ceph-node5
10.0.209.87         ceph-node6.ibm.com ceph-node6
10.0.208.89         ceph-node7.ibm.com ceph-node7
```

重要

セットアップに関係するすべてのノード (Ceph クラスター内のすべてのノードとすべての NFS クライアントノード) にドメイン名が存在することを確認してください。

手順

以下の手順に従って、KDC をインストールして設定します。すでに KDC をインストールして設定している場合は、この部分をスキップしてください。

1. KDC をセットアップするマシンに、必要な RPM がインストールされているかどうかを確認します。

```
[root@host ~]# rpm -qa | grep krb5

krb5-libs-1.20.1-9.el9_2.x86_64
krb5-pkinit-1.20.1-9.el9_2.x86_64
krb5-server-1.20.1-9.el9_2.x86_64
krb5-server-ldap-1.20.1-9.el9_2.x86_64
krb5-devel-1.20.1-9.el9_2.x86_64
krb5-workstation-1.20.1-9.el9_2.x86_64
```

-

注記

- Kerberos レルム名に従ったドメイン名を使用することを推奨します。たとえば、レルム - **PUNE.IBM.COM**、管理プリンシパル - **admin/admin** です。
- インストールされた設定ファイルを編集して、新しい KDC を反映するようにしてください。KDC は IP アドレスまたは DNS 名として指定できることに注意してください。

2. **krb5.conf** ファイルを更新します。

注記

krb5.conf ファイル内の **kdc** および **admin_server** IP を使用して、すべてのレルム (**default_realm** および **domain_realm**) を更新する必要があります。

```
[root@host ~]# cat /etc/krb5.conf

# To opt out of the system crypto-policies configuration of krb5, remove the
# symlink at /etc/krb5.conf.d/crypto-policies which will not be recreated.

includedir /etc/krb5.conf.d/
[logging]
    default = [FILE:/var/log/krb5libs.log](file:///var/log/krb5libs.log)
    kdc = [FILE:/var/log/krb5kdc.log](file:///var/log/krb5kdc.log)
    admin_server = [FILE:/var/log/kadmind.log](file:///var/log/kadmind.log)
[libdefaults]
    dns_lookup_realm = false
    ticket_lifetime = 24h
    renew_lifetime = 7d
    forwardable = true
    rdns = false
    pkinit_anchors = [FILE:/etc/pki/tls/certs/ca-bundle.crt](file:///etc/pki/tls/certs/ca-bundle.crt)
    spake_preauth_groups = edwards25519
    dns_canonicalize_hostname = fallback
    qualify_shortname = ""
    default_realm = PUNE.IBM.COM
    default_ccache_name = KEYRING:persistent:%{uid}
[realms]
    PUNE.IBM.COM = {
      kdc = 10.0.210.222:88
      admin_server = 10.0.210.222:749
    }
[domain_realm]
  .redhat.com = PUNE.IBM.COM
  redhat.com = PUNE.IBM.COM
```

3. **krb5.conf** ファイルを更新します。

注記

kdc.conf ファイル内のレルムを更新する必要があります。

```
[root@host ~]# cat /var/kerberos/krb5kdc/kdc.conf

 [kdcdefaults]
 kdc_ports = 88
 kdc_tcp_ports = 88
 spake_preauth_kdc_challenge = edwards25519
[realms]
   PUNE.IBM.COM = {
      master_key_type = aes256-cts-hmac-sha384-192
      acl_file = /var/kerberos/krb5kdc/kadm5.acl
      dict_file = /usr/share/dict/words
      default_principal_flags = +preauth
      admin_keytab = /var/kerberos/krb5kdc/kadm5.keytab
      supported_enctypes = aes256-cts-hmac-sha384-192:normal aes128-cts-hmac-sha256-
128:normal aes256-cts-hmac-sha1-96:normal aes128-cts-hmac-sha1-96:normal
camellia256-cts-cmac:normal camellia128-cts-cmac:normal arcfour-hmac-md5:normal
     # Supported encryption types for FIPS mode:
     #supported_enctypes = aes256-cts-hmac-sha384-192:normal aes128-cts-hmac-sha256-
128:normal
  }
```

4. **kdb5_util** を使用して KDC データベースを作成します。

注記

ホスト名が **DNS** または **/etc/hosts** 経由で解決できることを確認してください。

```
[root@host ~]# kdb5_util create -s -r [PUNE.IBM.COM](http://pune.ibm.com/)

Initializing database '/var/kerberos/krb5kdc/principal' for realm 'PUNE.IBM.COM',
master key name 'K/M@PUNE.IBM.COM'
You will be prompted for the database Master Password.
It is important that you NOT FORGET this password.
Enter KDC database master key:
Re-enter KDC database master key to verify:
```

5. 管理者を ACL ファイルに追加します。

```
[root@host ~]# cat /var/kerberos/krb5kdc/kadm5.acl

*/admin@PUNE.IBM.COM *
```

この出力は、管理インスタンスを持つ PUNE.IBM.COM レルム内のすべてのプリンシパルが、すべての管理者権限を持っていることを示します。

6. Kerberos データベースに管理者を追加します。

```
[root@host ~]# kadmin.local

Authenticating as principal root/admin@PUNE.IBM.COM with password.
kadmin.local:  addprinc admin/admin@PUNE.IBM.COM
No policy specified for admin/admin@PUNE.IBM.COM; defaulting to no policy
Enter password for principal "admin/admin@PUNE.IBM.COM":
```

```
Re-enter password for principal "admin/admin@PUNE.IBM.COM":
Principal "admin/admin@PUNE.IBM.COM" created.
kadmin.local:
```

7. **kdc** と **kadmind** を起動します。

```
# krb5kdc
# kadmind
```

検証

- **kdc** と **kadmind** が適切に実行されているかどうかを確認します。

```
# ps -eaf | grep krb

root     27836     1  0 07:35 ?        00:00:00 krb5kdc
root     27846 13956  0 07:35 pts/8    00:00:00 grep --color=auto krb
# ps -eaf | grep kad
root     27841     1  0 07:35 ?        00:00:00 kadmind
root     27851 13956  0 07:36 pts/8    00:00:00 grep --color=auto kad
```

- セットアップが正しいかどうかを確認します。

```
[root@host ~]# kinit admin/admin
Password for admin/admin@PUNE.IBM.COM:

[root@ceph-mani-o7fdxp-node4 ~]# klist
Ticket cache: KCM:0
Default principal: admin/admin@PUNE.IBM.COM

Valid starting     Expires            Service principal
10/25/23 06:37:08  10/26/23 06:37:01  krbtgt/PUNE.IBM.COM@PUNE.IBM.COM
    renew until 10/25/23 06:37:08
```

11.14.2. Kerberos クライアントのセットアップ

Kerberos クライアントマシンは KDC と同期する必要があります。NTP を使用して KDC とクライアントを確実に同期してください。時差が 5 分以上あると、Kerberos 認証が失敗し、clock skew エラーが出力されます。この手順は、Kerberos 認証に参加するすべてのシステム (NFS クライアントや NFS Ganesha コンテナーが実行されるホストなど) の前提条件です。

手順

1. 必要な RPM を確認します。

```
[root@host ~]# rpm -qa | grep krb5

krb5-libs-1.20.1-9.el9_2.x86_64
krb5-pkinit-1.20.1-9.el9_2.x86_64
krb5-workstation-1.20.1-9.el9_2.x86_64
```

2. KDC サーバーのものと同様の **krb5.conf** ファイルを更新します。

```
[root@host ~]# cat /etc/krb5.conf

# To opt out of the system crypto-policies configuration of krb5, remove the
# symlink at /etc/krb5.conf.d/crypto-policies which will not be recreated.

includedir /etc/krb5.conf.d/
[logging]
    default = [FILE:/var/log/krb5libs.log](file:///var/log/krb5libs.log)
    kdc = [FILE:/var/log/krb5kdc.log](file:///var/log/krb5kdc.log)
    admin_server = [FILE:/var/log/kadmind.log](file:///var/log/kadmind.log)
[libdefaults]
    dns_lookup_realm = false
    ticket_lifetime = 24h
    renew_lifetime = 7d
    forwardable = true
    rdns = false
    pkinit_anchors = [FILE:/etc/pki/tls/certs/ca-bundle.crt](file:///etc/pki/tls/certs/ca-bundle.crt)
    spake_preauth_groups = edwards25519
    dns_canonicalize_hostname = fallback
    qualify_shortname = ""
    default_realm = PUNE.IBM.COM
   default_ccache_name = KEYRING:persistent:%{uid}
[realms]
  PUNE.IBM.COM = {
     kdc = 10.0.210.222:88
     admin_server = 10.0.210.222:749
  }
[domain_realm]
  .IBM.com = PUNE.IBM.COM
  IBM.com = PUNE.IBM.COM
```

検証

- クライアント設定を検証します。

```
[root@host ~]# kinit admin/admin

Password for admin/admin@PUNE.IBM.COM:
[root@ceph-mani-o7fdxp-node5 ~]# klist
Ticket cache: KCM:0
Default principal: admin/admin@PUNE.IBM.COM

Valid starting     Expires            Service principal
10/25/23 08:49:12  10/26/23 08:49:08  krbtgt/PUNE.IBM.COM@PUNE.IBM.COM
  renew until 10/25/23 08:49:12
```

11.14.3. NFS 固有の Kerberos セットアップ

NFS サーバーとクライアントの両方のサービスプリンシパルを作成する必要があります。それぞれのキーは、キータブファイルに保存されます。これらのプリンシパルは、GSS_RPCSEC で必要な初期セキュリティーコンテキストをセットアップするために必要です。これらのサービスプリンシパルは、nfs/@REALM のような形式です。/etc/krb5.conf ファイルを稼働システムから Ceph ノードにコピーできます。

手順

手順

1. そのホストのサービスプリンシパルを作成します。

```
[root@host ~]# kinit admin/admin

Password for admin/admin@PUNE.IBM.COM:
[root@host ~]# kadmin
Authenticating as principal admin/admin@PUNE.IBM.COM with password.
Password for admin/admin@PUNE.IBM.COM:
kadmin:  addprinc -randkey nfs/<hostname>.ibm.com
No policy specified for nfs/<hostname>.ibm.com@PUNE.IBM.COM; defaulting to no policy
Principal "nfs/<hostname>.ibm.com@PUNE.IBM.COM" created.
```

2. キーをキータブファイルに追加します。

注記

このステップでは、ユーザーはすでに NFS サーバー上にいて、**kadmin** インターフェイスを使用しています。ここで、キータブの操作は NFS サーバーのキータブに反映されます。

```
kadmin:  ktadd nfs/<hostname>.ibm.com

Entry for principal nfs/<hostname>.ibm.com with kvno 2, encryption type aes256-cts-hmac-sha384-192 added to keytab [FILE:/etc/krb5.keytab](file:///etc/krb5.keytab).
Entry for principal nfs/<hostname>.ibm.com with kvno 2, encryption type aes128-cts-hmac-sha256-128 added to keytab [FILE:/etc/krb5.keytab](file:///etc/krb5.keytab).
Entry for principal nfs/<hostname>.ibm.com with kvno 2, encryption type aes256-cts-hmac-sha1-96 added to keytab [FILE:/etc/krb5.keytab](file:///etc/krb5.keytab).
Entry for principal nfs/<hostname>.ibm.com with kvno 2, encryption type aes128-cts-hmac-sha1-96 added to keytab [FILE:/etc/krb5.keytab](file:///etc/krb5.keytab).
Entry for principal nfs/<hostname>.ibm.com with kvno 2, encryption type camellia256-cts-cmac added to keytab [FILE:/etc/krb5.keytab](file:///etc/krb5.keytab).
Entry for principal nfs/<hostname>.ibm.com with kvno 2, encryption type camellia128-cts-cmac added to keytab [FILE:/etc/krb5.keytab](file:///etc/krb5.keytab).
Entry for principal nfs/<hostname>.ibm.com with kvno 2, encryption type arcfour-hmac added to keytab [FILE:/etc/krb5.keytab](file:///etc/krb5.keytab).
kadmin:
```

3. NFS Ganesha コンテナーが実行されているすべての Ceph ノードとすべての NFS クライアントで、手順 1 と 2 を実行します。

11.14.4. NFS Ganesha コンテナーの設定

以下の手順に従って、Ceph 環境で NFS Ganesha を設定します。

手順

1. 既存の NFS Ganesha コンテナー設定を取得します。

```
[ceph: root@host /]# ceph orch ls --service-type nfs --export

service_type: nfs
```

```
service_id: c_ganesha
service_name: nfs.c_ganesha
placement:
  hosts:
  - host1
  - host2
  - host3
spec:
  port: 2049
```

2. **/etc/krb5.conf** ファイルと **/etc/krb5.keytab`** ファイルをホストからコンテナーに渡すように、コンテナー設定を変更します。これらのファイルは、ランタイム時に NFS Ganesha が参照して、受信したサービスチケットを検証し、Ganesha と NFS クライアント (krb5p) 間の通信を保護します。

```
[root@host ~]# cat nfs.yaml

service_type: nfs
service_id: c_ganesha
service_name: nfs.c_ganesha
placement:
  hosts:
  - host1
  - host2
  - host3
spec:
  port: 2049
extra_container_args:
  - "-v"
  - "/etc/krb5.keytab:/etc/krb5.keytab:ro"
  - "-v"
  - "/etc/krb5.conf:/etc/krb5.conf:ro"
```

3. 変更した nfs.yaml ファイルを cephadm シェル内で使用できるようにします。

```
[root@host ~]# cephadm shell --mount nfs.yaml:/var/lib/ceph/nfs.yaml

Inferring fsid ff1c1498-73ec-11ee-af38-fa163e9a17fd
Inferring config /var/lib/ceph/ff1c1498-73ec-11ee-af38-fa163e9a17fd/mon.ceph-msaini-qp49z7-node1-installer/config
Using ceph image with id 'fada497f9c5f' and tag 'ceph-7.0-rhel-9-containers-candidate-73711-20231018030025' created on 2023-10-18 03:03:39 +0000 UTC
registry-proxy.engineering.ibm.com/rh-osbs/rhceph@sha256:e66e5dd79d021f3204a183f5dbe4537d0c0e4b466df3b2cc4d50cc79c0f34d75
```

4. ファイルに必要な変更が加えられているかどうかを検証します。

```
[ceph: root@host /]# cat /var/lib/ceph/nfs.yaml

service_type: nfs
service_id: c_ganesha
service_name: nfs.c_ganesha
placement:
  hosts:
```

```
  - host1
  - host2
  - host3
spec:
  port: 2049
extra_container_args:
  - "-v"
  - "/etc/krb5.keytab:/etc/krb5.keytab:ro"
  - "-v"
  - "/etc/krb5.conf:/etc/krb5.conf:ro"
```

5. 必要な変更を NFS Ganesha コンテナーに適用し、コンテナーを再デプロイします。

```
[ceph: root@host /]# ceph orch apply -i /var/lib/ceph/nfs.yaml

Scheduled nfs.c_ganesha update...
[ceph: root@ceph-msaini-qp49z7-node1-installer /]# ceph orch redeploy nfs.c_ganesha
Scheduled to redeploy nfs.c_ganesha.1.0.ceph-msaini-qp49z7-node1-installer.sxzuts on host
'ceph-msaini-qp49z7-node1-installer'
Scheduled to redeploy nfs.c_ganesha.2.0.ceph-msaini-qp49z7-node2.psuvki on host 'ceph-
msaini-qp49z7-node2'
Scheduled to redeploy nfs.c_ganesha.0.0.ceph-msaini-qp49z7-node3.qizzvk on host 'ceph-
msaini-qp49z7-node3'
```

6. 再デプロイされたサービスに必要な変更が加えられているかどうかを検証します。

```
[ceph: root@host /]# ceph orch ls --service-type nfs --export

service_type: nfs
service_id: c_ganesha
service_name: nfs.c_ganesha
placement:
  hosts:
  - ceph-msaini-qp49z7-node1-installer
  - ceph-msaini-qp49z7-node2
  - ceph-msaini-qp49z7-node3
extra_container_args:
- -v
- /etc/krb5.keytab:/etc/krb5.keytab:ro
- -v
- /etc/krb5.conf:/etc/krb5.conf:ro
spec:
  port: 2049
```

7. **krb5* (krb5, krb5i, krb5p)** セキュリティーフレーバーを持つようにエクスポート定義を変更します。

注記

上記のセットアップを完了した後、このようなエクスポートを作成できます。

```
[ceph: root@host /]# ceph nfs export info c_ganesha /exp1

{
```

```
  "access_type": "RW",
  "clients": [],
  "cluster_id": "c_ganesha",
  "export_id": 1,
  "fsal": {
    "fs_name": "fs1",
    "name": "CEPH",
    "user_id": "nfs.c_ganesha.1"
  },
  "path": "/volumes/_nogroup/exp1/81f9a67e-ddf1-4b5a-9fe0-d87effc7ca16",
  "protocols": [
    4
  ],
  "pseudo": "/exp1",
  "sectype": [
    "krb5"
  ],
  "security_label": true,
  "squash": "none",
  "transports": [
    "TCP"
  ]
}
```


11.14.5. NFS クライアントサイドのアクション

以下は、NFS クライアントが実行できる操作の一部です。

手順

1. サービスプリンシパルを作成します。

```
kadmin:  addprinc -randkey nfs/<hostname>.ibm.com@PUNE.IBM.COM
No policy specified for nfs/<hostname>.ibm.com@PUNE.IBM.COM; defaulting to no policy
Principal "nfs/<hostname>.ibm.com@PUNE.IBM.COM" created.
kadmin:  ktadd nfs/<hostname>.ibm.com@PUNE.IBM.COM
Entry for principal nfs/<hostname>.ibm.com@PUNE.IBM.COM with kvno 2, encryption type
aes256-cts-hmac-sha384-192 added to keytab [FILE:/etc/krb5.keytab](file:///etc/krb5.keytab).
Entry for principal nfs/<hostname>.ibm.com@PUNE.IBM.COM with kvno 2, encryption type
aes128-cts-hmac-sha256-128 added to keytab [FILE:/etc/krb5.keytab](file:///etc/krb5.keytab).
Entry for principal nfs/<hostname>.ibm.com@PUNE.IBM.COM with kvno 2, encryption type
aes256-cts-hmac-sha1-96 added to keytab [FILE:/etc/krb5.keytab](file:///etc/krb5.keytab).
Entry for principal nfs/<hostname>.ibm.com@PUNE.IBM.COM with kvno 2, encryption type
aes128-cts-hmac-sha1-96 added to keytab [FILE:/etc/krb5.keytab](file:///etc/krb5.keytab).
Entry for principal nfs/<hostname>.ibm.com@PUNE.IBM.COM with kvno 2, encryption type
camellia256-cts-cmac added to keytab [FILE:/etc/krb5.keytab](file:///etc/krb5.keytab).
Entry for principal nfs/<hostname>.ibm.com@PUNE.IBM.COM with kvno 2, encryption type
camellia128-cts-cmac added to keytab [FILE:/etc/krb5.keytab](file:///etc/krb5.keytab).
Entry for principal nfs/<hostname>.ibm.com@PUNE.IBM.COM with kvno 2, encryption type
arcfour-hmac added to keytab [FILE:/etc/krb5.keytab](file:///etc/krb5.keytab).
```

2. **rpc.gssd** サービスを再起動して、変更された/新しいキータブファイルを有効にします。

```
# systemctl restart rpc-gssd
```

3. NFS エクスポートをマウントします。

   構文

   ```
   [root@host ~]# mount -t nfs -o vers=4.1,port=2049 <IP>:/<export_name> >mount_point>
   ```

   例

   ```
   mount -t nfs -o vers=4.1,port=2049 10.8.128.233:/ganesha /mnt/test/
   ```

4. ユーザーを作成します。NFS エクスポートがマウントされると、マウントされたエクスポートを操作するために通常ユーザーが使用されます。これらの通常ユーザー (通常はシステム上のローカルユーザー、または LDAP/AD などの集中型システムからのユーザー) は、Kerberos セットアップの一部である必要があります。セットアップの種類に応じて、ローカルユーザーも KDC に作成する必要があります。

11.14.6. セットアップの検証

セットアップを検証するには、以下の手順に従ってください。

手順

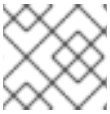
- Kerberos チケットを **使用せずに**、通常のユーザーとしてエクスポートにアクセスします。

  ```
  [user@host ~]$ klist

  klist: Credentials cache 'KCM:1001' not found

  [user@host ~]$ cd /mnt
  -bash: cd: /mnt: Permission denied
  ```

- Kerberos チケットを **使用して**、通常のユーザーとしてエクスポートにアクセスします。

  ```
  [user@host ~]$ kinit sachin

  Password for user@PUNE.IBM.COM:

  [user@host ~]$ klist
  Ticket cache: KCM:1001
  Default principal: user@PUNE.IBM.COM

  Valid starting     Expires            Service principal
  10/27/23 12:57:21  10/28/23 12:57:17  krbtgt/PUNE.IBM.COM@PUNE.IBM.COM
    renew until 10/27/23 12:57:21

  [user@host ~]$ cd /mnt

  [user@host mnt]$ klist

  Ticket cache: KCM:1001
  Default principal: user@PUNE.IBM.COM

  Valid starting     Expires            Service principal
  10/27/23 12:57:21  10/28/23 12:57:17  krbtgt/PUNE.IBM.COM@PUNE.IBM.COM
  ```

```
renew until 10/27/23 12:57:21
10/27/23 12:57:28  10/28/23 12:57:17  nfs/ceph-msaini-qp49z7-node1-installer.ibm.com@
renew until 10/27/23 12:57:21
Ticket server: nfs/ceph-msaini-qp49z7-node1-installer.ibm.com@PUNE.IBM.COM
```



注記

注記: **nfs/** service のチケットはクライアント上で監視されます。

第12章 SNMP トラップの設定

ストレージ管理者は、Red Hat Ceph Storage クラスターに簡易ネットワーク管理プロトコル (SNMP) ゲートウェイをデプロイおよび設定して、Prometheus Alertmanager からアラートを受信し、SNMP トラップとしてそれらをクラスターにルーティングできます。

12.1. 簡易ネットワーク管理プロトコル

簡易ネットワーク管理プロトコル (SNMP) は、さまざまなハードウェアおよびソフトウェアプラットフォームにまたがる分散システムおよびデバイスを監視するためのもので、最も広く使用されているオープンプロトコルの 1 つです。Ceph の SNMP 統合は、Prometheus Alertmanager クラスターからゲートウェイデーモンへのアラートの転送に重点を置いています。ゲートウェイデーモンはアラートを SNMP 通知に変換し、これを指定された SNMP 管理プラットフォームに送信します。ゲートウェイデーモンは **snmp_notifier_project** からのものであり、認証と暗号化による SNMP V2c および V3 サポートを提供します。

Red Hat Ceph Storage SNMP ゲートウェイサービスは、デフォルトでゲートウェイの 1 つのインスタンスをデプロイします。配置情報を提供することで、これを増やすことができます。ただし、複数の SNMP ゲートウェイデーモンを有効にすると、SNMP 管理プラットフォームは同じイベントに対して複数の通知を受け取ります。

SNMP トラップはアラートメッセージであり、Prometheus Alertmanager はこれらのアラートを SNMP 通知機能に送信し、SNMP 通知機能は指定されたアラートのラベルでオブジェクト識別子 (OID) を探します。各 SNMP トラップには一意の ID があり、ステータスが更新された追加のトラップを特定の SNMP ポーラーに送信できます。SNMP は Ceph の健全性チェックにフックして、すべての健全性警告が特定の SNMP トラップを生成するようにします。

正しく動作し、デバイスステータスに関する情報をユーザーに転送して監視するために、SNMP はいくつかのコンポーネントに依存しています。SNMP を設定する 4 つの主要なコンポーネントがあります。

- **SNMP マネージャー** - SNMP マネージャーは、管理ステーションとも呼ばれ、ネットワーク監視プラットフォームを実行するコンピューターです。SNMP 対応デバイスをポーリングし、それらからデータを取得するロールを持つプラットフォーム。SNMP マネージャーは、エージェントにクエリーを実行し、エージェントからの応答を受信し、エージェントからの非同期イベントを確認します。

- **SNMP エージェント** - SNMP エージェントは、マネージドのシステムで実行されるプログラムであり、システムの MIB データベースが含まれています。これらは、帯域幅やディスク容量などのデータを収集して集約し、Management Information Base (MIB) に送信します。

- **Management information base (MIB)**- これらは SNMP エージェントに含まれるコンポーネントです。SNMP マネージャーはこれをデータベースとして使用し、エージェントに特定の情報へのアクセスを要求します。この情報は、ネットワーク管理システム (NMS) に必要です。NMS は、エージェントをポーリングしてこれらのファイルから情報を取得してから、それらをユーザーが見ることができるグラフやディスプレイに変換する処理を行います。MIB には、ネットワークデバイスによって決定される統計値と制御値が含まれています。

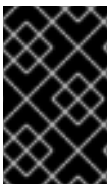
- **SNMP デバイス**

以下のバージョンの SNMP は互換性があり、ゲートウェイの実装でサポートされています。

- **V2c** - 認証なしでコミュニティー文字列を使用し、外部からの攻撃に対して脆弱です。

- **V3 authNoPriv** - 暗号化せずにユーザー名とパスワードの認証を使用します。

- **V3 authPriv** - SNMP 管理プラットフォームへの暗号化を伴うユーザー名とパスワードの認証を使用します。



重要

SNMP トラップを使用する場合は、バージョン番号のセキュリティー設定が正しいことを確認して、SNMP に固有の脆弱性を最小限に抑え、許可されていないユーザーからネットワークを保護してください。

12.2. SNMPTRAPD の設定

snmptrapd デーモンには、**snmp-gateway** サービスの作成時に指定する必要のある認証設定が含まれているため、**snmp-gateway** をデプロイする前に Simple Network Management Protocol (SNMP) ターゲットを設定することが重要です。

SNMP ゲートウェイ機能は、Prometheus スタックで生成されたアラートを SNMP 管理プラットフォームに公開する手段を提供します。**snmptrapd** ツールに基づいて、宛先への SNMP トラップを設定できます。このツールを使用すると、1 つ以上の SNMP トラップリスナーを確立できます。

設定には以下のパラメーターが重要となります。

- **engine-id** は、デバイスの一意の識別子 (16 進数) であり、SNMPV3 ゲートウェイで必要とされています。Red Hat では、このパラメーターに `8000C53F_CLUSTER_FSID_WITHOUT_DASHES_` を使用することを推奨しています。
- **SNMP_COMMUNITY_FOR_SNMPV2** パラメーターである **snmp-community** は、SNMPV2c ゲートウェイに対して **public** です。
- **AUTH_PROTOCOL** である **auth-protocol** は、SNMPV3 ゲートウェイでは必須であり、デフォルトでは **SHA** になります。
- SNMPV3 ゲートウェイでは、**PRIVACY_PROTOCOL** である **privacy-protocol** は必須となります。
- **PRIVACY_PASSWORD** は、暗号化された SNMPV3 ゲートウェイでは必須となります。
- **SNMP_V3_AUTH_USER_NAME** はユーザー名であり、SNMPV3 ゲートウェイでは必須となります。
- **SNMP_V3_AUTH_PASSWORD** はパスワードであり、SNMPV3 ゲートウェイでは必須となります。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- ノードへの root レベルのアクセス。
- Red Hat Enterprise Linux システムに **firewalld** をインストールします。

手順

1. SNMP 管理ホストで、SNMP パッケージをインストールします。

   例

   -

```
[root@host01 ~]# dnf install -y net-snmp-utils net-snmp
```

2. SNMP のポート 162 を開いて、アラートを受信します。

   例

   ```
   [root@host01 ~]# firewall-cmd --zone=public --add-port=162/udp
   [root@host01 ~]# firewall-cmd --zone=public --add-port=162/udp --permanent
   ```

3. 管理情報ベース (MIB) を実装して、SNMP 通知を理解し、宛先ホストでの SNMP サポートを強化します。メインリポジトリーから raw ファイルをコピーします: https://github.com/ceph/ceph/blob/master/monitoring/snmp/CEPH-MIB.txt

   例

   ```
   [root@host01 ~]# curl -o CEPH_MIB.txt -L
   https://raw.githubusercontent.com/ceph/ceph/master/monitoring/snmp/CEPH-MIB.txt
   [root@host01 ~]# scp CEPH_MIB.txt root@host02:/usr/share/snmp/mibs
   ```

4. **snmptrapd** ディレクトリーを作成します。

   例

   ```
   [root@host01 ~]# mkdir /root/snmptrapd/
   ```

5. SNMP バージョンに基づいて、各プロトコルの **snmptrapd** ディレクトリーに設定ファイルを作成します。

   構文

   ```
   format2 %V\n% Agent Address: %A \n Agent Hostname: %B \n Date: %H - %J - %K - %L - %M - %Y \n Enterprise OID: %N \n Trap Type: %W \n Trap Sub-Type: %q \n Community/Infosec Context: %P \n Uptime: %T \n Description: %W \n PDU Attribute/Value Pair Array:\n%v \n -------------- \n
   createuser -e 0x_ENGINE_ID_ SNMPV3_AUTH_USER_NAME AUTH_PROTOCOL SNMP_V3_AUTH_PASSWORD PRIVACY_PROTOCOL PRIVACY_PASSWORD
   authuser log,execute SNMP_V3_AUTH_USER_NAME
   authCommunity   log,execute,net SNMP_COMMUNITY_FOR_SNMPV2
   ```

   - SNMPV2c の場合、以下のように **snmptrapd_public.conf** ファイルを作成します。

     例

     ```
     format2 %V\n% Agent Address: %A \n Agent Hostname: %B \n Date: %H - %J - %K - %L - %M - %Y \n Enterprise OID: %N \n Trap Type: %W \n Trap Sub-Type: %q \n Community/Infosec Context: %P \n Uptime: %T \n Description: %W \n PDU Attribute/Value Pair Array:\n%v \n -------------- \n

     authCommunity   log,execute,net public
     ```

     ここでの **public** 設定は、**snmp-gateway** サービスをデプロイするときに使用される **snmp_community** 設定と一致する必要があります。

- 認証のみの SNMPV3 の場合は、以下のように **snmptrapd_auth.conf** ファイルを作成します。

  例

  ```
  format2 %V\n% Agent Address: %A \n Agent Hostname: %B \n Date: %H - %J - %K - %L - %M - %Y \n Enterprise OID: %N \n Trap Type: %W \n Trap Sub-Type: %q \n Community/Infosec Context: %P \n Uptime: %T \n Description: %W \n PDU Attribute/Value Pair Array:\n%v \n -------------- \n
  createuser -e 0x8000C53Ff64f341c655d11eb8778fa163e914bcc myuser SHA mypassword
  authuser log,execute myuser
  ```

  0x8000C53Ff64f341c655d11eb8778fa163e914bcc 文字列は **engine_id** で、**myuser** と **mypassword** は認証情報です。パスワードのセキュリティーは、**SHA** アルゴリズムによって定義されます。

  これは、**snmp-gateway** デーモンをデプロイするための設定に対応します。

  例

  ```
  snmp_v3_auth_username: myuser
  snmp_v3_auth_password: mypassword
  ```

- 認証と暗号化を使用する SNMPV3 の場合、以下のように **snmptrapd_authpriv.conf** ファイルを作成します。

  例

  ```
  format2 %V\n% Agent Address: %A \n Agent Hostname: %B \n Date: %H - %J - %K - %L - %M - %Y \n Enterprise OID: %N \n Trap Type: %W \n Trap Sub-Type: %q \n Community/Infosec Context: %P \n Uptime: %T \n Description: %W \n PDU Attribute/Value Pair Array:\n%v \n -------------- \n
  createuser -e 0x8000C53Ff64f341c655d11eb8778fa163e914bcc myuser SHA mypassword DES mysecret
  authuser log,execute myuser
  ```

  0x8000C53Ff64f341c655d11eb8778fa163e914bcc 文字列は **engine_id** で、**myuser** と **mypassword** は認証情報です。パスワードセキュリティーは **SHA** アルゴリズムで定義され、**DES** はプライバシー暗号化のタイプになります。

  これは、**snmp-gateway** デーモンをデプロイするための設定に対応します。

  例

  ```
  snmp_v3_auth_username: myuser
  snmp_v3_auth_password: mypassword
  snmp_v3_priv_password: mysecret
  ```

6. SNMP 管理ホストでデーモンを実行します。

構文

```
/usr/sbin/snmptrapd -M /usr/share/snmp/mibs -m CEPH-MIB.txt -f -C -c
/root/snmptrapd/CONFIGURATION_FILE -Of -Lo :162
```

例

```
[root@host01 ~]# /usr/sbin/snmptrapd -M /usr/share/snmp/mibs -m CEPH-MIB.txt -f -C -c
/root/snmptrapd/snmptrapd_auth.conf -Of -Lo :162
```

7. ストレージクラスターでアラートがトリガーされた場合は、SNMP 管理ホストで出力を監視できます。SNMP トラップと、MIB によってデコードされたトラップを確認します。

例

```
NET-SNMP version 5.8
 Agent Address: 0.0.0.0
 Agent Hostname: <UNKNOWN>
 Date: 15 - 5 - 12 - 8 - 10 - 4461391
 Enterprise OID: .
 Trap Type: Cold Start
 Trap Sub-Type: 0
 Community/Infosec Context: TRAP2, SNMP v3, user myuser, context
 Uptime: 0
 Description: Cold Start
 PDU Attribute/Value Pair Array:
.iso.org.dod.internet.mgmt.mib-2.1.3.0 = Timeticks: (292276100) 3 days, 19:52:41.00
.iso.org.dod.internet.snmpV2.snmpModules.1.1.4.1.0 = OID:
.iso.org.dod.internet.private.enterprises.ceph.cephCluster.cephNotifications.prometheus.promMgr.promMgrPrometheusInactive
.iso.org.dod.internet.private.enterprises.ceph.cephCluster.cephNotifications.prometheus.promMgr.promMgrPrometheusInactive.1 = STRING:
"1.3.6.1.4.1.50495.1.2.1.6.2[alertname=CephMgrPrometheusModuleInactive]"
.iso.org.dod.internet.private.enterprises.ceph.cephCluster.cephNotifications.prometheus.promMgr.promMgrPrometheusInactive.2 = STRING: "critical"
.iso.org.dod.internet.private.enterprises.ceph.cephCluster.cephNotifications.prometheus.promMgr.promMgrPrometheusInactive.3 = STRING: "Status: critical
- Alert: CephMgrPrometheusModuleInactive
  Summary: Ceph's mgr/prometheus module is not available
  Description: The mgr/prometheus module at 10.70.39.243:9283 is unreachable. This could mean that the module has been disabled or the mgr itself is down.
Without the mgr/prometheus module metrics and alerts will no longer function. Open a shell to ceph and use 'ceph -s' to determine whether the mgr is active. If the mgr is not active, restart it, otherwise you can check the mgr/prometheus module is loaded with 'ceph mgr module ls'  and if it's not listed as enabled, enable it with 'ceph mgr module enable prometheus'"
```

上記の例では、Prometheus モジュールが無効になった後にアラートが生成されます。

関連情報

- **Red Hat Ceph Storage オペレーションガイド** の SNMP ゲートウェイのデプロイ セクションを参照してください。

12.3. SNMP ゲートウェイのデプロイ

SNMPV2c または SNMPV3 のいずれかを使用して、Simple Network Management Protocol (SNMP) ゲートウェイをデプロイできます。SNMP ゲートウェイをデプロイするには、次の 2 つの方法があります。

1. 認証情報ファイルを作成する方法
2. すべての詳細を含む 1 つのサービス設定 yaml ファイルを作成する方法。

以下のパラメーターを使用して、バージョンに基づいて SNMP ゲートウェイをデプロイできます。

- **service_type** は **snmp-gateway** です。
- **service_name** は、任意のユーザー定義の文字列です。
- **count** は、ストレージクラスターにデプロイされる SNMP ゲートウェイの数です。
- **snmp_destination** パラメーターは、hostname:port の形式である必要があります。
- **engine-id** は、デバイスの一意の識別子 (16 進数) であり、SNMPV3 ゲートウェイで必要とされています。Red Hat では、このパラメーターに `8000C53F_CLUSTER_FSID_WITHOUT_DASHES_` を使用することを推奨しています。
- **snmp_community** パラメーターは、SNMPV2c ゲートウェイに対して **public** です。
- **auth-protocol** は SNMPV3 ゲートウェイでは必須であり、デフォルトでは **SHA** になります。
- **privacy-protocol** では、認証と暗号化を備えた SNMPV3 ゲートウェイが必須となります。
- デフォルトでは、ポートは **9464** です。
- シークレットとパスワードをオーケストレーターに渡すには、**-i FILENAME** を指定する必要があります。

SNMP ゲートウェイサービスがデプロイまたは更新されると、Prometheus Alertmanager 設定が自動的に更新され、objectidentifier を持つアラートが SNMP ゲートウェイデーモンに転送されてさらに処理されます。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- ノードへの root レベルのアクセス。
- SNMP 管理ホストである宛先ホストで **snmptrapd** を設定します。

手順

1. Cephadm シェルにログインします。

   例

   ```
   [root@host01 ~]# cephadm shell
   ```

2. SNMP ゲートウェイをデプロイする必要があるホストのラベルを作成します。

   構文

```
ceph orch host label add HOSTNAME snmp-gateway
```

例

```
[ceph: root@host01 /]# ceph orch host label add host02 snmp-gateway
```

3. SNMP のバージョンに応じて、認証情報ファイルまたはサービス設定ファイルを作成します。

- SNMPV2c の場合、以下のようにファイルを作成します。

  例

  ```
  [ceph: root@host01 /]# cat snmp_creds.yml

  snmp_community: public
  ```

  OR

  例

  ```
  [ceph: root@host01 /]# cat snmp-gateway.yml

  service_type: snmp-gateway
  service_name: snmp-gateway
  placement:
    count: 1
  spec:
    credentials:
      snmp_community: public
    port: 9464
    snmp_destination: 192.168.122.73:162
    snmp_version: V2c
  ```

- 認証のみの SNMPV3 の場合は、以下のようにファイルを作成します。

  例

  ```
  [ceph: root@host01 /]# cat snmp_creds.yml

  snmp_v3_auth_username: myuser
  snmp_v3_auth_password: mypassword
  ```

  OR

  例

  ```
  [ceph: root@host01 /]# cat snmp-gateway.yml

  service_type: snmp-gateway
  service_name: snmp-gateway
  placement:
    count: 1
  spec:
  ```

```
  credentials:
    snmp_v3_auth_password: mypassword
    snmp_v3_auth_username: myuser
  engine_id: 8000C53Ff64f341c655d11eb8778fa163e914bcc
  port: 9464
  snmp_destination: 192.168.122.1:162
  snmp_version: V3
```

- 認証と暗号化を使用する SNMPV3 の場合、以下のようにファイルを作成します。

  例

```
[ceph: root@host01 /]# cat snmp_creds.yml

snmp_v3_auth_username: myuser
snmp_v3_auth_password: mypassword
snmp_v3_priv_password: mysecret
```

  OR

  例

```
[ceph: root@host01 /]# cat snmp-gateway.yml

service_type: snmp-gateway
service_name: snmp-gateway
placement:
  count: 1
spec:
  credentials:
    snmp_v3_auth_password: mypassword
    snmp_v3_auth_username: myuser
    snmp_v3_priv_password: mysecret
  engine_id: 8000C53Ff64f341c655d11eb8778fa163e914bcc
  port: 9464
  snmp_destination: 192.168.122.1:162
  snmp_version: V3
```

4. **ceph orch** コマンドを実行します。

   構文

```
ceph orch apply snmp-gateway --snmp_version=V2c_OR_V3 --
destination=SNMP_DESTINATION [--port=PORT_NUMBER]\
[--engine-id=8000C53F_CLUSTER_FSID_WITHOUT_DASHES_] [--auth-
protocol=MDS_OR_SHA] [--privacy_protocol=DES_OR_AES] -i FILENAME
```

   OR

   構文

```
ceph orch apply -i FILENAME.yml
```

- SNMPV2c の場合、**snmp_creds** ファイルを使用して、**snmp-version** を **V2c** として指定して、**ceph orch** コマンドを実行します。

  例

  ```
  [ceph: root@host01 /]# ceph orch apply snmp-gateway --snmp-version=V2c --destination=192.168.122.73:162 --port=9464 -i snmp_creds.yml
  ```

- SNMPV3 で認証のみの場合、**snmp_creds** ファイルを用いて、**snmp-version** を **V3** および **engine-id** として指定して、**ceph orch** コマンドを実行します。

  例

  ```
  [ceph: root@host01 /]# ceph orch apply snmp-gateway --snmp-version=V3 --engine-id=8000C53Ff64f341c655d11eb8778fa163e914bcc--destination=192.168.122.73:162 -i snmp_creds.yml
  ```

- 認証と暗号化を使用する SNMPV3 の場合、**snmp_creds** ファイルを使用して、**snmp-version** を **V3**、**privacy-protocol**、および **engine-id** として指定して、**ceph orch** コマンドを実行します。

  例

  ```
  [ceph: root@host01 /]# ceph orch apply snmp-gateway --snmp-version=V3 --engine-id=8000C53Ff64f341c655d11eb8778fa163e914bcc--destination=192.168.122.73:162 --privacy-protocol=AES -i snmp_creds.yml
  ```

  OR

- すべての SNMP バージョンの場合、**snmp-gateway** ファイルを使用して、以下のコマンドを実行します。

  例

  ```
  [ceph: root@host01 /]# ceph orch apply -i snmp-gateway.yml
  ```

関連情報

- Red Hat Ceph Storage **オペレーションガイド** の `snmptrapd` の設定 のセクションを参照してください。

第13章 ノードの障害の処理

ストレージクラスター内でノード全体に障害が発生する可能性があります。ストレージ管理者が行うノード障害の処理は、ディスク障害の処理と同様です。ノードの障害として Ceph が 1 つのディスクに対してのみ配置グループ (PG) を復元する代わりに、そのノード内のディスクのすべての PG を復元する必要があります。Ceph は OSD がすべてダウンしていることを検出し、自己修復として知られる復元プロセスを自動的に開始します。

ノードの障害シナリオは 3 つあります。

- ノードの置き換えには、失敗したノードから root ディスクおよび Ceph OSD ディスクを使用します。
- ノードを置き換え、オペレーティングシステムを再インストールし、障害が発生したノードから Ceph OSD ディスクを使用します。
- ノードを置き換え、オペレーティングシステムを再インストールし、すべての新規 Ceph OSD ディスクを使用します。

各ノードの置き換えに関するシナリオのワークフローの概要は、**Workflow for replacing a node** (https://access.redhat.com/documentation/ja-jp/red_hat_ceph_storage/7/html-single/operations_guide/#ops_workflow-for replacing-a-node) を参照してください。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- 障害のあるノード。

13.1. ノードの追加または削除前の考慮事項

Ceph の未処理の機能の 1 つは、ランタイム時に Ceph OSD ノードを追加または削除できる機能です。つまり、ストレージクラスターの容量のサイズを変更したり、ストレージクラスターを縮小せずにハードウェアを置き換えることができることを意味します。

ストレージクラスターの状態が劣化 (**degraded**) している間に Ceph クライアントを提供する機能にも運用上の利点があります。たとえば、残業や週末ではなく、通常の営業時間内にハードウェアを追加、削除、または交換できます。ただし、Ceph OSD ノードの追加および削除により、パフォーマンスに大きな影響を与える可能性があります。

Ceph OSD ノードを追加または削除する前に、ストレージクラスターのパフォーマンスへの影響を考慮してください。

- ストレージクラスターの容量を拡張または縮小するか、Ceph OSD ノードを追加または削除することで、ストレージクラスターのリバランスとしてバックフィルを予測します。このリバランス期間中に、Ceph は追加のリソースを使用します。これにより、ストレージクラスターのパフォーマンスに影響する可能性があります。
- 実稼働用 Ceph Storage クラスターでは、Ceph OSD ノードに特定のタイプのストレージストラテジーを容易にする特定のハードウェア設定があります。
- Ceph OSD ノードは CRUSH 階層の一部であるため、ノードの追加または削除のパフォーマンスへの影響は通常 CRUSH ルールセットを使用するプールのパフォーマンスに影響します。

関連情報

- 詳細は、Red Hat Ceph Storage の **ストレージストラテジーガイド** を参照してください。

13.2. ノードを交換するワークフロー

ノードの障害シナリオは 3 つあります。ノードを交換するときは、シナリオごとに次の概要ワークフローを使用します。

- **障害が発生したノードの root ディスクと Ceph OSD ディスクを使用してノードを交換する**
- **オペレーティングシステムを再インストールし、障害が発生したノードの Ceph OSD ディスクを使用してノードを交換する**
- **オペレーティングシステムを再インストールし、すべての新しい Ceph OSD ディスクを使用してノードを交換する**

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- 障害のあるノード。

13.2.1. 障害が発生したノードの root ディスクと Ceph OSD ディスクを使用してノードを交換する

障害が発生したノードの root ディスクと Ceph OSD ディスクを使用してノードを交換します。

手順

1. バックフィルを無効にします。

   構文

   ```
   ceph osd set noout
   ceph osd set noscrub
   ceph osd set nodeep-scrub
   ```

   例

   ```
   [ceph: root@host01 /]# ceph osd set noout
   [ceph: root@host01 /]# ceph osd set noscrub
   [ceph: root@host01 /]# ceph osd set nodeep-scrub
   ```

2. ノードを置き換え、古いノードからディスクを取得し、それらを新規ノードに追加します。

3. バックフィルを有効にします。

   構文

   ```
   ceph osd unset noout
   ceph osd unset noscrub
   ceph osd unset nodeep-scrub
   ```

   例

```
[ceph: root@host01 /]# ceph osd unset noout
[ceph: root@host01 /]# ceph osd unset noscrub
[ceph: root@host01 /]# ceph osd unset nodeep-scrub
```

13.2.2. オペレーティングシステムを再インストールし、障害が発生したノードの Ceph OSD ディスクを使用してノードを交換する

オペレーティングシステムを再インストールし、障害が発生したノードの Ceph OSD ディスクを使用してノードを交換します。

手順

1. バックフィルを無効にします。

   構文

   ```
   ceph osd set noout
   ceph osd set noscrub
   ceph osd set nodeep-scrub
   ```

   例

   ```
   [ceph: root@host01 /]# ceph osd set noout
   [ceph: root@host01 /]# ceph osd set noscrub
   [ceph: root@host01 /]# ceph osd set nodeep-scrub
   ```

2. Ceph 設定のバックアップを作成します。

   構文

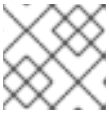
   ```
   cp /etc/ceph/ceph.conf /PATH_TO_BACKUP_LOCATION/ceph.conf
   ```

   例

   ```
   [ceph: root@host01 /]# cp /etc/ceph/ceph.conf /some/backup/location/ceph.conf
   ```

3. ノードを置き換え、障害が発生したノードから Ceph OSD ディスクを追加します。

4. ディスクを JBOD として設定します。

   注記

   これはストレージ管理者が行う必要があります。

5. オペレーティングシステムをインストールします。オペレーティングシステム要件の詳細は、**Red Hat Ceph Storage のオペレーティングシステム要件**を参照してください。オペレーティングシステムのインストールの詳細は、Red Hat Enterprise Linux 製品ドキュメント を参照してください。

注記

これはシステム管理者が行う必要があります。

6. Ceph の設定を復元します。

構文

```
cp /PATH_TO_BACKUP_LOCATION/ceph.conf /etc/ceph/ceph.conf
```

例

```
[ceph: root@host01 /]# cp /some/backup/location/ceph.conf /etc/ceph/ceph.conf
```

7. Ceph Orchestrator コマンドを使用して、新しいノードをストレージクラスターに追加します。Ceph デーモンは、それぞれのノードに自動的に配置されます。詳細は、Ceph OSD ノードの追加 を参照してください。

8. バックフィルを有効にします。

構文

```
ceph osd unset noout
ceph osd unset noscrub
ceph osd unset nodeep-scrub
```

例

```
[ceph: root@host01 /]# ceph osd unset noout
[ceph: root@host01 /]# ceph osd unset noscrub
[ceph: root@host01 /]# ceph osd unset nodeep-scrub
```

13.2.3. オペレーティングシステムを再インストールし、すべての新しい Ceph OSD ディスクを使用してノードを交換する

オペレーティングシステムを再インストールし、すべての新しい Ceph OSD ディスクを使用してノードを置き換えます。

手順

1. バックフィルを無効にします。

構文

```
ceph osd set noout
ceph osd set noscrub
ceph osd set nodeep-scrub
```

例

```
[ceph: root@host01 /]# ceph osd set noout
[ceph: root@host01 /]# ceph osd set noscrub
[ceph: root@host01 /]# ceph osd set nodeep-scrub
```

2. 障害のあるノードのすべての OSD をストレージクラスターから削除します。詳細は、**Ceph OSD ノードの削除** を参照してください。

3. Ceph 設定のバックアップを作成します。

   構文

   ```
   cp /etc/ceph/ceph.conf /PATH_TO_BACKUP_LOCATION/ceph.conf
   ```

   例

   ```
   [ceph: root@host01 /]# cp /etc/ceph/ceph.conf /some/backup/location/ceph.conf
   ```

4. ノードを置き換え、障害が発生したノードから Ceph OSD ディスクを追加します。

5. ディスクを JBOD として設定します。

   注記

   これはストレージ管理者が行う必要があります。

6. オペレーティングシステムをインストールします。オペレーティングシステム要件の詳細は、**Red Hat Ceph Storage のオペレーティングシステム要件**を参照してください。オペレーティングシステムのインストールの詳細は、Red Hat Enterprise Linux 製品ドキュメント を参照してください。

   注記

   これはシステム管理者が行う必要があります。

7. Ceph Orchestrator コマンドを使用して、新しいノードをストレージクラスターに追加します。Ceph デーモンは、それぞれのノードに自動的に配置されます。詳細は、**Ceph OSD ノードの追加** を参照してください。

8. バックフィルを有効にします。

   構文

   ```
   ceph osd unset noout
   ceph osd unset noscrub
   ceph osd unset nodeep-scrub
   ```

   例

   ```
   [ceph: root@host01 /]# ceph osd unset noout
   [ceph: root@host01 /]# ceph osd unset noscrub
   [ceph: root@host01 /]# ceph osd unset nodeep-scrub
   ```

13.3. パフォーマンスに関する考慮事項

以下の要素は通常、Ceph OSD ノードの追加時または削除時のストレージクラスターのパフォーマンスに影響します。

- Ceph クライアントは、Ceph への I/O インターフェイスの負荷を配置します。つまり、クライアントはプールに負荷を置きます。プールは CRUSH ルールセットにマップします。基礎となる CRUSH 階層により、Ceph は障害ドメインにデータを配置できます。基礎となる Ceph OSD ノードにクライアント負荷が高いプールが必要な場合、クライアントの負荷は復元時間に大きな影響を及ぼし、パフォーマンスが低下する可能性があります。書き込み操作には持続性のためにデータのレプリケーションが必要になるため、特に書き込み集約型クライアント負荷は、ストレージクラスターを回復するのに要する時間が長くなる可能性があります。

- 通常、追加または削除している容量は、クラスターの復元に影響を及ぼします。さらに、追加または削除するノードのストレージの密度も復元時間に影響を及ぼす可能性があります。たとえば、36 OSD を持つノードは、通常、12 OSD を持つノードよりも復元にかかる時間が長くなります。

- ノードを削除する際には、十分な容量を確保して、**完全な比率** または **ほぼ完全比率** に到達しないようにします。ストレージクラスターが **フル比率** になると、Ceph は書き込み動作を一時停止してデータの損失を防ぎます。

- Ceph OSD ノードは、少なくとも 1 つの Ceph CRUSH 階層にマッピングし、階層は少なくとも 1 つのプールにマップされます。CRUSH ルールセットを使用する各プールは、Ceph OSD ノードが追加または削除される際にパフォーマンスに影響が出ます。

- レプリケーションプールは、データのディープコピーを複製するネットワーク帯域幅を使用する傾向がありますが、イレイジャーコーディングプールはより多くの CPU を使用して **k+m** コーディングのチャンクを計算する傾向があります。データに存在するより多くのコピーで、ストレージクラスターを回復するのにかかる時間が長くなります。たとえば、大きなプールまたは **k+m** チャンクの数が大きい場合は、同じデータのコピー数が少ないレプリケーションプールよりも復元にかかる時間が長くなります。

- ドライブ、コントローラー、およびネットワークインターフェイスカードはすべて、復元時間に影響を与える可能性があるスループットの特性を持ちます。一般に、10 Gbps や SSD などのスループット特性が高いノードは、1Gbps や SATA ドライブなどのスループット特性が低いノードよりも迅速に回復します。

13.4. ノードの追加または削除に関する推奨事項

Red Hat は、ノード内の一度に 1 つの OSD を追加または削除して、次の OSD に進む前にストレージクラスターを回復させることを推奨します。これは、ストレージクラスターのパフォーマンスへの影響を最小限に抑えるのに役立ちます。ノードに障害が発生した場合は、一度に 1 つの OSD ではなく、ノード全体を一度に変更する必要がある場合があります。

OSD を削除するには、以下を実行します。

- **Ceph Orchestrator を使用した OSD デーモンの削除**

OSD を追加するには、以下を実行します。

- **利用可能なすべてのデバイスへの Ceph OSD のデプロイ**
- **高度なサービス仕様を使用した Ceph OSD のデプロイ**
- **特定のデバイスおよびホストへの Ceph OSD のデプロイ**

Ceph OSD ノードを追加または削除する場合は、他の継続中のプロセスがストレージクラスターのパフォーマンスにも影響することを検討してください。クライアント I/O への影響を減らすために、Red Hat では以下を推奨します。

容量の計算

Ceph OSD ノードを削除する前に、ストレージクラスターがそのすべての OSD の内容を **完全な比率** に到達せずにバックフィルするようにしてください。**フル比率** に達すると、ストレージクラスターは書き込み操作を拒否するようになります。

スクラビングを一時的に無効にする

スクラビングはストレージクラスターのデータの持続性を確保するために不可欠ですが、リソース集約型です。Ceph OSD ノードを追加または削除する前に、スクラビングおよびディープスクラビングを無効にして、現在のスクラビング操作を完了してから続行します。

```
ceph osd set noscrub
ceph osd set nodeep-scrub
```

Ceph OSD ノードを追加または削除すると、ストレージクラスターが **active+clean** 状態に戻り、**noscrub** および **nodeep-scrub** の設定を解除します。

```
ceph osd unset noscrub
ceph osd unset nodeep-scrub
```

バックフィルと復元の制限

妥当なデータの永続性がある場合は、パフォーマンスが劣化 (**degraded**) した状態での動作に問題はありません。たとえば、**osd_pool_default_size = 3** および **osd_pool_default_min_size = 2** を使用してストレージクラスターを操作できます。可能な限り早い復元時間用にストレージクラスターを調整することができますが、これにより Ceph クライアントの I/O パフォーマンスが大幅に影響を受ける可能性があります。最大の Ceph クライアント I/O パフォーマンスを維持するには、バックフィルと復元の操作を制限し、その操作に長い時間がかかる可能性があります。

```
osd_max_backfills = 1
osd_recovery_max_active = 1
osd_recovery_op_priority = 1
```

osd_recovery_sleep などの sleep パラメーターおよび delay パラメーターを設定することもできます。

配置グループの数を増やす

最後に、ストレージクラスターのサイズを拡張する場合は、配置グループの数を増やすことが必要となる場合があります。配置グループの数を拡張する必要がある場合、Red Hat はプレースメントグループの数を段階的に増やすことを推奨します。配置グループの数を大幅に増やすと、パフォーマンスが大幅に低下します。

13.5. CEPH OSD ノードの追加

Red Hat Ceph Storage クラスターの容量を拡張するには、OSD ノードを追加します。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。

- ネットワーク接続が割り当てられたプロビジョニングされたノード

手順

1. ストレージクラスターの他のノードが、短縮ホスト名で新規ノードに到達できることを確認します。
2. スクラビングを一時的に無効にします。

   例

   ```
   [ceph: root@host01 /]# ceph osd set noscrub
   [ceph: root@host01 /]# ceph osd set nodeep-scrub
   ```

3. バックフィルおよび復元機能を制限します。

   構文

   ```
   ceph tell DAEMON_TYPE.* injectargs --OPTION_NAME VALUE [--OPTION_NAME VALUE]
   ```

   例

   ```
   [ceph: root@host01 /]# ceph tell osd.* injectargs --osd-max-backfills 1 --osd-recovery-max-active 1 --osd-recovery-op-priority 1
   ```

4. クラスターの SSH 公開鍵をフォルダーにデプロイメントします。

   構文

   ```
   ceph cephadm get-pub-key > ~/PATH
   ```

   例

   ```
   [ceph: root@host01 /]# ceph cephadm get-pub-key > ~/ceph.pub
   ```

5. ceph クラスターの SSH 公開鍵を、新たなホストの root ユーザーの **authorized_keys** ファイルにコピーします。

   構文

   ```
   ssh-copy-id -f -i ~/PATH root@HOST_NAME_2
   ```

   例

   ```
   [ceph: root@host01 /]# ssh-copy-id -f -i ~/ceph.pub root@host02
   ```

6. 新規ノードを CRUSH マップに追加します。

   構文

```
ceph orch host add NODE_NAME IP_ADDRESS
```

例

```
[ceph: root@host01 /]# ceph orch host add host02 10.10.128.70
```

7. ノードの各ディスクの OSD をストレージクラスターに追加します。
   - **利用可能なすべてのデバイスへの Ceph OSD のデプロイ**
   - **高度なサービス仕様を使用した Ceph OSD のデプロイ**
   - **特定のデバイスおよびホストへの Ceph OSD のデプロイ**

重要

OSD ノードを Red Hat Ceph Storage クラスターに追加する場合、Red Hat は、1 度に 1 つの OSD デーモンを追加し、次の OSD に進む前にクラスターを **active+clean** 状態に復元できるようにすることを推奨します。

関連情報

- 詳細は、**Red Hat Ceph Storage 設定ガイド** の**実行時の特定の設定の設定** セクションを参照してください。
- CRUSH 階層の適切な場所にノードを配置するための詳細は、Red Hat Ceph Storage の **ストレージストラテジーガイド**の **バケットの追加** および **バケットの移動** セクションを参照してください。

13.6. CEPH OSD ノードの削除

ストレージクラスターの容量を減らすには、OSD ノードを削除します。

警告

Ceph OSD ノードを削除する前に、ストレージクラスターが **完全な比率** に到達せずにすべての OSD の内容をバックフィルするようにしてください。**フル比率** に達すると、ストレージクラスターは書き込み操作を拒否するようになります。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- ストレージクラスター内のすべてのノードへの root レベルのアクセス。

手順

1. ストレージクラスターの容量を確認します。

構文

```
ceph df
rados df
ceph osd df
```

2. スクラビングを一時的に無効にします。

構文

```
ceph osd set noscrub
ceph osd set nodeep-scrub
```

3. バックフィルおよび復元機能を制限します。

構文

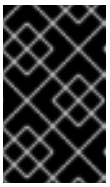
```
ceph tell DAEMON_TYPE.* injectargs --OPTION_NAME VALUE [--OPTION_NAME VALUE]
```

例

```
[ceph: root@host01 /]# ceph tell osd.* injectargs --osd-max-backfills 1 --osd-recovery-max-active 1 --osd-recovery-op-priority 1
```

4. ノード上の各 OSD をストレージクラスターから削除します。

- Ceph Orchestrator を使用した OSD デーモンの削除



重要

ストレージクラスターから OSD ノードを削除する場合、Red Hat は、ノード内の一度に 1 つの OSD を削除してから、次の OSD を削除する前にクラスターが **active+clean** 状態に回復できるようにすることを推奨します。

a. OSD を削除したら、ストレージクラスターが **ほぼ完全比率** に達していないことを確認します。

構文

```
ceph -s
ceph df
```

b. ノードのすべての OSD がストレージクラスターから削除されるまでこの手順を繰り返します。

5. すべての OSD が削除されたら、ホストを削除します。

- Ceph Orchestrator を使用したホストの削除。

関連情報

- 詳細は、**Red Hat Ceph Storage 設定ガイド**の **ライタイムでの特定の設定の設定** セクションを参照してください。

13.7. ノードの障害のシミュレーション

ハードノードの障害をシミュレーションするには、ノードの電源をオフにし、オペレーティングシステムを再インストールします。

前提条件

- 正常かつ実行中の Red Hat Ceph Storage クラスター
- ストレージクラスター内のすべてのノードへの root レベルのアクセス。

手順

1. ストレージクラスターの容量を確認し、ノードの削除への影響を確認します。

   例

   ```
   [ceph: root@host01 /]# ceph df
   [ceph: root@host01 /]# rados df
   [ceph: root@host01 /]# ceph osd df
   ```

2. 必要に応じて、復元およびバックフィルを無効にします。

   例

   ```
   [ceph: root@host01 /]# ceph osd set noout
   [ceph: root@host01 /]# ceph osd set noscrub
   [ceph: root@host01 /]# ceph osd set nodeep-scrub
   ```

3. ノードをシャットダウンします。
4. ホスト名を変更する場合は、CRUSH マップからノードを削除します。

   例

   ```
   [ceph: root@host01 /]# ceph osd crush rm host03
   ```

5. ストレージクラスターのステータスを確認します。

   例

   ```
   [ceph: root@host01 /]# ceph -s
   ```

6. ノードにオペレーティングシステムを再インストールします。
7. 新しいノードを追加します。
   - **Ceph Orchestrator を使用したホストの追加**
8. 必要に応じて、復元およびバックフィルを有効にします。

例

```
[ceph: root@host01 /]# ceph osd unset noout
[ceph: root@host01 /]# ceph osd unset noscrub
[ceph: root@host01 /]# ceph osd unset nodeep-scrub
```

9. Ceph のヘルスを確認します。

例

```
[ceph: root@host01 /]# ceph -s
```

関連情報

- 詳細は、Red Hat Ceph Storage インストールガイドを参照してください。

第14章 データセンター障害の処理

ストレージ管理者は、データセンター障害を回避するために予防措置を取ることができます。これには、以下の予防措置があります。

- データセンターインフラストラクチャーの設定
- CRUSH マップ階層内に障害ドメインの設定
- ドメイン内での障害ノードの指定

前提条件

- 正常かつ実行中の Red Hat Ceph Storage クラスター
- ストレージクラスター内のすべてのノードへの root レベルのアクセス。

14.1. データセンター障害の回避

データセンターインフラストラクチャーの設定

ストレッチクラスター内の各データセンターには、ローカル機能および依存関係を反映するために異なるストレージクラスターの設定を指定できます。データの保存に役立つデータセンター間でレプリケーションを設定します。1つのデータセンターに障害が発生しても、ストレージクラスターの他のデータセンターにデータのコピーが含まれます。

CRUSH マップ階層内での障害ドメインの設定

障害またはフェイルオーバーのドメインは、ストレージクラスター内のドメインの冗長コピーです。アクティブなドメインが失敗すると、障害ドメインはアクティブドメインになります。

デフォルトで、CRUSH マップはフラット階層内のストレージクラスターのすべてのノードをリスト表示します。ただし、最善の結果を得るには、CRUSH マップ内に論理階層構造を作成します。階層は、各ノードが属するドメインと、障害のあるドメインを含む、ストレージクラスター内のそれらのドメイン間の関係を指定します。階層内の各ドメインの障害ドメインを定義すると、ストレージクラスターの信頼性が向上します。

複数のデータセンターを含むストレージクラスターを計画する際には、CRUSH マップ階層内にノードを配置するため、1つのデータセンターが停止した場合には、残りのストレージクラスターは稼働し続けます。

ドメイン内での障害ノードの設計

ストレージクラスター内のデータに 3 方向のレプリケーションを使用する予定の場合には、障害ドメイン内のノードの場所を考慮してください。データセンター内で停止が発生した場合は、一部のデータが1つのコピーにのみ存在する可能性があります。このシナリオでは、2 つのオプションがあります。

- 標準設定で、データは読み取り専用ステータスのままにします。
- ライブは、停止期間に1つのコピーのみを行います。

標準設定では、ノード間でのデータ配置のランダム性のため、すべてのデータが影響を受けるわけではありませんが、一部のデータは1つのコピーしか持つことができず、ストレージクラスターは読み取り専用モードに戻ります。ただし、一部のデータが1つのコピーにのみ存在する場合、ストレージクラスターは読み取り専用モードに戻ります。

14.2. データセンター障害の処理

Red Hat Ceph Storage は、ストレッチクラスターでデータセンターのいずれかを失うなど、インフラストラクチャーに非常に致命的な障害がある場合があります。標準のオブジェクトストアのユースケースでは、3 つのデータセンターすべての設定は、それらの間にレプリケーションを設定して個別に実行できます。このシナリオでは、各データセンターのストレージクラスター設定は異なり、ローカルの機能と依存関係を反映する可能性があります。

配置階層の論理構造を考慮する必要があります。適切な CRUSH マップは使用でき、インフラストラクチャー内の障害ドメインの階層構造が反映されます。論理階層定義を使用すると、標準の階層定義を使用することではなく、ストレージクラスターの信頼性が向上します。障害ドメインは CRUSH マップで定義されます。デフォルトの CRUSH マップには、フラットな階層のすべてのノードが含まれます。ストレッチクラスターなどの 3 つのデータセンター環境では、ノードの配置は、1 つのデータセンターが停止できるように管理する必要がありますが、ストレージクラスターは稼働したままです。データに対して 3 方向レプリケーションを使用する場合に、ノードがある障害について検討してください。

以下の例では、作成されるマップは 6 つの OSD ノードを持つストレージクラスターの初期設定から派生しています。この例では、すべてのノードが 1 つのディスクを持つため、1 つの OSD しかありません。すべてのノードは、階層ツリーの標準 **root** であるデフォルトの **root** 下に分類されます。2 つの OSD に重みが割り当てられているため、これらの OSD は他の OSD よりも少ないデータチャンクを受け取ります。これらのノードは、最初の OSD ディスクよりも大きなディスクを持つ後から導入されました。これは、ノードのグループが失敗しているデータ配置には影響しません。

例

```
[ceph: root@host01 /]# ceph osd tree
ID WEIGHT  TYPE NAME          UP/DOWN REWEIGHT PRIMARY-AFFINITY
-1 0.33554 root default
-2 0.04779     host host03
 0 0.04779         osd.0          up  1.00000          1.00000
-3 0.04779     host host02
 1 0.04779         osd.1          up  1.00000          1.00000
-4 0.04779     host host01
 2 0.04779         osd.2          up  1.00000          1.00000
-5 0.04779     host host04
 3 0.04779         osd.3          up  1.00000          1.00000
-6 0.07219     host host06
 4 0.07219         osd.4          up  0.79999          1.00000
-7 0.07219     host host05
 5 0.07219         osd.5          up  0.79999          1.00000
```

論理階層定義を使用してノードを同じデータセンターにグループ化すると、データの配置の成熟度を実行できます。**root**、**datacenter**、**rack**、**row**、および **host** の定義タイプにより、3 つのデータセンターのストッククラスターで障害ドメインを反映させることができます。

- ノード host01 および host02 はデータセンター 1 (DC1) にあります。
- ノード host03 および host05 はデータセンター 2 (DC2) にあります。
- ノード host04 および host06 はデータセンター 3 (DC3) にあります。
- すべてのデータセンターが同じ構造に属する (全 DC)

ホストのすべての OSD はホスト定義に属しているため、変更は必要ありません。その他のすべての割り当ては、ストレージクラスターの実行時に以下によって調整できます。

- 以下のコマンドで **バケット** 構造を定義します。

```
ceph osd crush add-bucket allDC root
ceph osd crush add-bucket DC1 datacenter
ceph osd crush add-bucket DC2 datacenter
ceph osd crush add-bucket DC3 datacenter
```

- CRUSH マップを変更して、ノードをこの構造内の適切な場所に移動します。

```
ceph osd crush move DC1 root=allDC
ceph osd crush move DC2 root=allDC
ceph osd crush move DC3 root=allDC
ceph osd crush move host01 datacenter=DC1
ceph osd crush move host02 datacenter=DC1
ceph osd crush move host03 datacenter=DC2
ceph osd crush move host05 datacenter=DC2
ceph osd crush move host04 datacenter=DC3
ceph osd crush move host06 datacenter=DC3
```

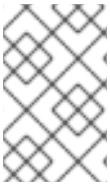
この構造内で、新しいホストや新しいディスクを追加することもできます。OSD を階層の右側に配置することにより、CRUSH アルゴリズムが冗長な部分を構造内の異なる障害ドメインに配置するように変更されます。

上記の例は、以下のようになります。

例

```
[ceph: root@host01 /]# ceph osd tree
ID  WEIGHT  TYPE NAME             UP/DOWN REWEIGHT PRIMARY-AFFINITY
 -8 6.00000 root allDC
 -9 2.00000     datacenter DC1
 -4 1.00000         host host01
  2 1.00000             osd.2          up  1.00000          1.00000
 -3 1.00000         host host02
  1 1.00000             osd.1          up  1.00000          1.00000
-10 2.00000     datacenter DC2
 -2 1.00000         host host03
  0 1.00000             osd.0          up  1.00000          1.00000
 -7 1.00000         host host05
  5 1.00000             osd.5          up  0.79999          1.00000
-11 2.00000     datacenter DC3
 -6 1.00000         host host06
  4 1.00000             osd.4          up  0.79999          1.00000
 -5 1.00000         host host04
  3 1.00000             osd.3          up  1.00000          1.00000
 -1       0 root default
```

上記のリストには、osd ツリーを表示することで、生成される CRUSH マップが表示されます。ホストがデータセンターにどのように属し、すべてのデータセンターが同じトップレベル構造に属しているかがわかりやすくなりましたが、場所が明確に区別されています。



注記

マップに応じてデータを適切な場所に配置すると、正常なクラスター内でのみ適切に機能します。一部の OSD が利用できない状況では、置き違えが発生する可能性があります。この誤差は、可能な場合は自動的に修正されます。

関連情報

- 詳細は、Red Hat Ceph Storage ストレージストラテジーガイドの CRUSH 管理 の章を参照してください。