Red Hat Ceph Storage 4

オペレーションガイド

Red Hat Ceph Storage の操作タスク

Last Updated: 2021-09-13

Red Hat Ceph Storage 4 オペレーションガイド

Red Hat Ceph Storage の操作タスク

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

法律上の通知

概要

本書では、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、弊社の CTO、Chris Wright のメッセージを参照してください。

目次

第1章 ストレージクラスターのサイズの管理

ストレージ管理者は、ストレージ容量が拡張または縮小する際に Ceph Monitor または OSD を追加または削除することにより、ストレージクラスターのサイズを管理できます。Ceph Ansible を使用するか、コマンドラインインターフェース (CLI) を使用して、ストレージクラスターのサイズを管理できます。

1.1. 前提条件

- 実行中の Red Hat Ceph Storage クラスター
- Ceph Monitor および OSD ノードへのルートレベルのアクセス

1.2. 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。

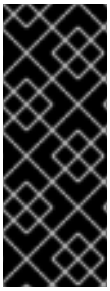
重要

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 モニターは、偶数のモニターよりも障害に対する回復性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 では、同じノードで Ceph Monitor と OSD を共存させるサポートはありません。これを行うと、ストレージクラスターのパフォーマンスに悪影響を与える可能性があります。

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

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 でサポートされる構成」を参照してください。

1.2.1. 新規 Ceph Monitor ノードの準備

デプロイメント用の新たな Ceph Monitor ノードを準備する前に、『Red Hat Ceph Storage インストールガイド』の「**Red Hat Ceph Storage のインストール要件**」の章を確認してください。

重要

新規 Ceph Monitor を別のノードにデプロイして、ストレージクラスター内のすべての Ceph Monitor ノードが同じハードウェアで実行される必要があります。

前提条件

- ネットワーク接続
- 新しいノードへのルートレベルのアクセス。

手順

1. 新規ノードをサーバーラックに追加します。
2. 新しいノードをネットワークに接続します。
3. Red Hat Enterprise Linux 7 または Red Hat Enterprise Linux 8 の最新バージョンをインストールします。
   a. Red Hat Enterprise Linux 7 の場合は、**ntp** をインストールし、信頼できるタイムソースを設定します。

   ```
   [root@mon ~]# yum install ntp
   ```

   b. Red Hat Enterprise Linux 8 の場合は、**chrony** をインストールし、信頼できるタイムソースを設定します。

   ```
   [root@mon ~]# dnf install chrony
   ```

4. ファイアウォールを使用している場合は、TCP ポート 6789 を開きます。

```
[root@mon ~]# firewall-cmd --zone=public --add-port=6789/tcp
[root@mon ~]# firewall-cmd --zone=public --add-port=6789/tcp --permanent
```

関連情報

- **chrony** の詳細は、「Red Hat Enterprise Linux 8 の基本システム設定の構成」を参照してください。

1.2.2. Ansible を使用した Ceph Monitor の追加

Red Hat は、奇数のモニターを維持するために、一度に 2 つの Ceph Monitor を追加することを推奨します。たとえば、ストレージクラスターに Ceph Monitor が 3 つある場合に、Red Hat はモニター数を 5 に増やすことを推奨します。

前提条件

- 新しいノードへのルートレベルのアクセス。
- Ansible 管理ノード。
- Ansible によりデプロイされた実行中の Red Hat Ceph Storage クラスター

手順

1. **[mons]** セクションの下に、新しい Ceph Monitor ノードを **/etc/ansible/hosts** Ansible インベントリーファイルに追加します。

   例

   ```
   [mons]
   monitor01
   monitor02
   monitor03
   NEW_MONITOR_NODE_NAME
   NEW_MONITOR_NODE_NAME
   ```

2. Ansible が Ceph ノードと通信できることを確認します。

   ```
   [root@admin ~]# ansible all -m ping
   ```

3. ディレクトリーを Ansible 設定ディレクトリーに移動します。

   ```
   [root@admin ~]# cd /usr/share/ceph-ansible
   ```

4. **ベアメタル** と **コンテナー** の両方のデプロイメントに、以下の Ansible Playbook を実行します。

   ```
   [root@admin ceph-ansible]# ansible-playbook infrastructure-playbooks/add-mon.yml -i hosts
   ```

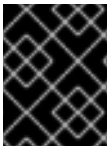
   Ansible Playbook の実行が終了すると、新しい Ceph Monitor ノードがストレージクラスターに表示されます。

関連情報

- Ansible インベントリー設定の詳細は、**{storage_product} インストールガイド** の「Ansible のインベントリーの場所の設定」セクションを参照してください。

1.2.3. コマンドラインインターフェースを使用した Ceph Monitor の追加

Red Hat は、奇数のモニターを維持するために、一度に 2 つの Ceph Monitor を追加することを推奨します。たとえば、ストレージクラスターに Ceph Monitor が 3 つある場合に、Red Hat はモニター数を 5 に増やすことを推奨します。



重要

Red Hat は、ノードごとに Ceph Monitor デーモンを 1 つだけ実行することを推奨します。

前提条件

- 実行中の Red Hat Ceph Storage クラスター
- 実行中の Ceph Monitor ノードへのルートレベルのアクセスと、新しいモニターノードへのアクセス。

手順

1. Red Hat Ceph Storage 4 Monitor リポジトリーを追加にします。

   Red Hat Enterprise Linux 7

   ```
   [root@mon ~]# subscription-manager repos --enable=rhel-7-server-rhceph-4-mon-rpms
   ```

   Red Hat Enterprise Linux 8

   ```
   [root@mon ~]# subscription-manager repos --enable=rhceph-4-mon-for-rhel-8-x86_64-rpms
   ```

2. **ceph-mon** パッケージを新しい Ceph Monitor ノードにインストールします。

   Red Hat Enterprise Linux 7

   ```
   [root@mon ~]# yum install ceph-mon
   ```

   Red Hat Enterprise Linux 8

   ```
   [root@mon ~]# dnf install ceph-mon
   ```

3. ストレージクラスター内の実行中のノードの Ceph 設定ファイルの **[mon]** セクションで **mon_host** 設定の一覧を編集します。

   a. 新規 Ceph Monitor ノードの IP アドレスを **mon_host** 設定一覧に追加します。

      構文

      ```
      [mon]
      mon_host = MONITOR_IP : PORT MONITOR_IP : PORT ... NEW_MONITOR_IP : PORT
      ```

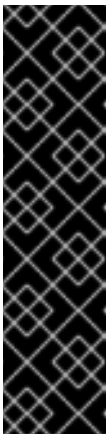
      Ceph 設定ファイルの [mon] セクションに新しい Ceph Monitor の IP アドレスを追加する代わりに、新規モニターノード用にファイルに特定のセクションを作成することができます。

      構文

```
[mon.MONITOR_ID]
host = MONITOR_ID
mon_addr = MONITOR_IP
```

注記

mon_host 設定の一覧は、DNS で解決できるホスト名または IP アドレスの一覧で、「,」、「;」、または「 」で区切ります。この一覧は、ストレージクラスターが起動または再起動時に新規の Monitor ノードを識別できるようにします。



重要

mon_initial_members 設定は、Ceph Monitor ノードの初期クォーラムグループを一覧表示します。そのグループの 1 つのメンバーが失敗すると、そのグループの別のノードが初期モニターノードになります。実稼働ストレージクラスターの高可用性を確保するには、Ceph 設定ファイルの **mon_initial_members** セクションおよび **mon_host** セクションに、少なくとも 3 つの監視ノードを一覧表示します。これにより、最初のモニターノードに障害が発生した場合にストレージクラスターをロックできなくなります。追加するモニターノードが **mon_initial_members** および **mon_host** の一部であるモニターを置き換える場合は、新しいモニターを両方のセクションに追加します。

4. 最初のクォーラムグループのモニター部分を作成するには、Ceph 設定ファイルの **[global]** セクションの **mon_initial_members** パラメーターにホスト名を追加します。

例

```
[global]
mon_initial_members = node1 node2 node3 node4 node5
...
[mon]
mon_host = 192.168.0.1:6789 192.168.0.2:6789 192.168.0.3:6789 192.168.0.4:6789
192.168.0.5:6789
...
[mon.node4]
host = node4
mon_addr = 192.168.0.4

[mon.node5]
host = node5
mon_addr = 192.168.0.5
```

5. 更新された Ceph 設定ファイルをすべての Ceph ノードおよび Ceph クライアントにコピーします。

構文

```
scp /etc/ceph/CLUSTER_NAME.conf TARGET_NODE_NAME:/etc/ceph
```

例

```
[root@mon ~]# scp /etc/ceph/ceph.conf node4:/etc/ceph
```

6. モニターのデータのディレクトリーを新規モニターノードに作成します。

   構文

   ```
   mkdir /var/lib/ceph/mon/CLUSTER_NAME - MONITOR_ID
   ```

   例

   ```
   [root@mon ~]# mkdir /var/lib/ceph/mon/ceph-node4
   ```

7. 実行中の Ceph Monitor ノードおよび新しいモニターノードに一時ディレクトリーを作成し、この手順に必要なファイルをこれらのディレクトリーで保持します。各ノードの一時ディレクトリーは、ノードのデフォルトディレクトリーとは異なる必要があります。これは、すべての手順の完了後に削除できます。

   構文

   ```
   mkdir TEMP_DIRECTORY_PATH_NAME
   ```

   例

   ```
   [root@mon ~]# mkdir /tmp/ceph
   ```

8. 実行中の Ceph Monitor ノードから新しい Ceph Monitor ノードに admin キーをコピーし、**ceph** コマンドを実行できるようにします。

   構文

   ```
   scp /etc/ceph/CLUSTER_NAME.client.admin.keyring TARGET_NODE_NAME:/etc/ceph
   ```

   例

   ```
   [root@mon ~]# scp /etc/ceph/ceph.client.admin.keyring node4:/etc/ceph
   ```

9. 実行中の Ceph Monitor ノードから、モニターキーリングを取得します。

   構文

   ```
   ceph auth get mon. -o TEMP_DIRECTORY_PATH_NAME/KEY_FILE_NAME
   ```

   例

   ```
   [root@mon ~]# ceph auth get mon. -o /tmp/ceph/ceph_keyring.out
   ```

10. 実行中の Ceph Monitor ノードから、モニターマップを取得します。

    構文

```
ceph mon getmap -o TEMP_DIRECTORY_PATH_NAME/MONITOR_MAP_FILE
```

例

```
[root@mon ~]# ceph mon getmap -o /tmp/ceph/ceph_mon_map.out
```

11. 収集した Ceph Monitor データを新しい Ceph Monitor ノードにコピーします。

構文

```
scp /tmp/ceph TARGET_NODE_NAME:/tmp/ceph
```

例

```
[root@mon ~]# scp /tmp/ceph node4:/tmp/ceph
```

12. 先に収集したデータから、新しいモニター用にデータディレクトリーを準備します。モニターからクォーラム情報を取得するため、モニターマップへのパスを「fsid」と共に指定します。モニターキーリングへのパスを指定します。

構文

```
ceph-mon -i MONITOR_ID --mkfs --monmap
TEMP_DIRECTORY_PATH_NAME/MONITOR_MAP_FILE --keyring
TEMP_DIRECTORY_PATH_NAME/KEY_FILE_NAME
```

例

```
[root@mon ~]# ceph-mon -i node4 --mkfs --monmap /tmp/ceph/ceph_mon_map.out --
keyring /tmp/ceph/ceph_keyring.out
```

13. カスタム名を持つストレージクラスターの場合は、以下の行を **/etc/sysconfig/ceph** ファイルに追加します。

構文

```
echo "CLUSTER=CUSTOM_CLUSTER_NAME" >> /etc/sysconfig/ceph
```

例

```
[root@mon ~]# echo "CLUSTER=example" >> /etc/sysconfig/ceph
```

14. 新規モニターノードで所有者およびグループのパーミッションを更新します。

構文

```
chown -R OWNER : GROUP DIRECTORY_PATH
```

例

```
[root@mon ~]# chown -R ceph:ceph /var/lib/ceph/mon
[root@mon ~]# chown -R ceph:ceph /var/log/ceph
[root@mon ~]# chown -R ceph:ceph /var/run/ceph
[root@mon ~]# chown -R ceph:ceph /etc/ceph
```

15. 新しい monitor ノードで **ceph-mon** プロセスを有効にして起動します。

構文

```
systemctl enable ceph-mon.target
systemctl enable ceph-mon@MONITOR_ID
systemctl start ceph-mon@MONITOR_ID
```

例

```
[root@mon ~]# systemctl enable ceph-mon.target
[root@mon ~]# systemctl enable ceph-mon@node4
[root@mon ~]# systemctl start ceph-mon@node4
```

関連情報

- 『Red Hat Ceph Storage インストールガイド』の「Red Hat Ceph Storage リポジトリーの有効化」セクションを参照してください。

1.2.4. 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一覧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。ピアに対して各モニターによって提供される接続スコアを評価し、最も接続されたモニターと信頼できる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。

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 1、2、または 3 に変更します。

構文

```
$ ceph mon set election_strategy {1|2|3}
```

- **classic** の場合は “1”
- **disallow** の場合は “2”
- **connectivity** の場合は “3”

1.2.5. Ansible を使用した Ceph Monitor の削除

Ansible で Ceph Monitor を削除するには、Playbook の **shrink-mon.yml** を使用します。

前提条件

- Ansible 管理ノード。
- Ansible によりデプロイされた実行中の Red Hat Ceph Storage クラスター

手順

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

```
[user@admin ~]$ cd /usr/share/ceph-ansible
```

2. **ベアメタル** および **コンテナー** のデプロイメントに、Ansible Playbook の **shrink-mon.yml** を実行します。

   構文

```
ansible-playbook infrastructure-playbooks/shrink-mon.yml -e mon_to_kill=NODE_NAME -u ANSIBLE_USER_NAME -i hosts
```

   以下を置き換えます。

   - **NODE_NAME** は、Ceph Monitor ノードの短いホスト名に置き換えます。Playbook の実行時に 1 つの Ceph Monitor は 1 つだけ削除できます。
   - **ANSIBLE_USER_NAME** は、Ansible ユーザーの名前に置き換えてください。

   例

```
[user@admin ceph-ansible]$ ansible-playbook infrastructure-playbooks/shrink-mon.yml -e mon_to_kill=monitor1 -u user -i hosts
```

3. ストレージクラスター内のすべての Ceph 設定ファイルから Ceph Monitor エントリーを削除します。
4. Ceph Monitor が正常に削除されていることを確認します。

```
[root@mon ~]# ceph -s
```

関連情報

- Red Hat Ceph Storage のインストールに関する詳細は、『Red Hat Ceph Storage インストールガイド』を参照してください。

- Ansible インベントリー設定の詳細は、**{storage_product} インストールガイド** の「Ansible のインベントリーの場所の設定」セクションを参照してください。

1.2.6. コマンドラインインターフェースを使用した Ceph Monitor の削除

Ceph Monitor を削除するには、ストレージクラスターから **ceph-mon** デーモンを削除し、ストレージクラスターマップを更新します。

前提条件

- 実行中の Red Hat Ceph Storage クラスター
- monitor ノードへのルートレベルのアクセス。

手順

1. Ceph Monitor サービスを停止します。

   構文

   ```
   systemctl stop ceph-mon@MONITOR_ID
   ```

   例

   ```
   [root@mon ~]# systemctl stop ceph-mon@node3
   ```

2. ストレージクラスターから Ceph Monitor を削除します。

   構文

   ```
   ceph mon remove MONITOR_ID
   ```

   例

   ```
   [root@mon ~]# ceph mon remove node3
   ```

3. Ceph 設定ファイルから Ceph Monitor エントリーを削除します。設定ファイルのデフォルトの場所は **/etc/ceph/ceph.conf** です。

4. Ceph 設定ファイルを、ストレージクラスターの残りの全 Ceph ノードに再配布します。

   構文

   ```
   scp /etc/ceph/CLUSTER_NAME.conf USER_NAME @ TARGET_NODE_NAME :/etc/ceph/
   ```

   例

   ```
   [root@mon ~]# scp /etc/ceph/ceph.conf root@node3:/etc/ceph/
   ```

5. **コンテナー** のデプロイメントの場合は、Ceph Monitor サービスを無効にし、削除します。

   a. Ceph Monitor サービスを無効にします。

構文

```
systemctl disable ceph-mon@MONITOR_ID
```

例

```
[root@mon ~]# systemctl disable ceph-mon@node3
```

b. **systemd** からサービスを削除します。

```
[root@mon ~]# rm /etc/systemd/system/ceph-mon@.service
```

c. **systemd** マネージャー設定を再読み込みします。

```
[root@mon ~]# systemctl daemon-reload
```

d. 障害が発生した Ceph Monitor ノードの状態をリセットします。

```
[root@mon ~]# systemctl reset-failed
```

e. **ceph-mon** パッケージを削除します。

Red Hat Enterprise Linux 7

```
[root@mon ~]# docker exec node3 yum remove ceph-mon
```

Red Hat Enterprise Linux 8

```
[root@mon ~]# podman exec node3 yum remove ceph-mon
```

6. 必要に応じて、Ceph Monitor データをアーカイブします。

構文

```
mv /var/lib/ceph/mon/CLUSTER_NAME - MONITOR_ID /var/lib/ceph/mon/removed-
CLUSTER_NAME - MONITOR_ID
```

例

```
[root@mon ~]# mv /var/lib/ceph/mon/ceph-node3 /var/lib/ceph/mon/removed-ceph-node3
```

7. 必要に応じて、Ceph Monitor データを削除します。

構文

```
rm -r /var/lib/ceph/mon/CLUSTER_NAME - MONITOR_ID
```

例

```
[root@mon ~]# rm -r /var/lib/ceph/mon/ceph-node3
```

1.2.7. 異常なストレージクラスターからの Ceph Monitor の削除

この手順では、正常でないストレージクラスターから **ceph-mon** デーモンを削除します。配置グループが **active + clean** にならない、正常でないストレージクラスター。

前提条件

- 実行中の Red Hat Ceph Storage クラスター
- Ceph Monitor ノードへのルートレベルのアクセス。
- Ceph Monitor ノードが少なくとも 1 台実行している。

手順

1. 存続する Ceph Monitor ノードにログインします。

   構文

   ```
   ssh root@MONITOR_NODE_NAME
   ```

2. **ceph-mon** デーモンを停止し、**monmap** ファイルのコピーを抽出します。

   構文

   ```
   systemctl stop ceph-mon@MONITOR_ID
   ceph-mon -i MONITOR_ID --extract-monmap TEMP_PATH
   ```

   例

   ```
   [root@mon ~]# systemctl stop ceph-mon@mon1
   [root@mon ~]# ceph-mon -i node1 --extract-monmap /tmp/monmap
   ```

3. Ceph Monitor 以外を削除します。

   構文

   ```
   monmaptool TEMPORARY_PATH --rm _MONITOR_ID
   ```

   例

   ```
   [root@mon ~]# monmaptool /tmp/monmap --rm node2
   ```

4. 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 MONITOR_ID --inject-monmap TEMP_PATH
   ```

   例

```
[root@mon ~]# ceph-mon -i node1 --inject-monmap /tmp/monmap
```

1.3. 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** デーモンで構成されます。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。

1.3.1. 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」**の章を確認してください。

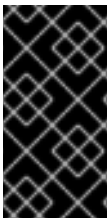
関連情報

- 詳しい情報は、**『Red Hat Ceph Storage 戦略ガイド』**を参照してください。*

1.3.2. コンテナーの OSD ID のドライブへのマッピング

場合によっては、コンテナー化された OSD が使用しているドライブを特定する必要がある場合があります。たとえば、OSD に問題がある場合には、ドライブのステータスを検証するために使用するドライブを把握しなければならない場合があります。また、コンテナー化されていない OSD の場合は、

OSD ID を参照して開始および停止しますが、コンテナー化された OSD を開始および停止するには、使用するドライブを参照します。



重要

以下の例は、Red Hat Enterprise Linux 8 で実行しています。Red Hat Enterprise Linux 8 では、**podman** がデフォルトのサービスとなり、古い **docker** サービスに置き換えられています。Red Hat Enterprise Linux 7 で実行している場合は、**podman** を **docker** に置き換え、指定したコマンドを実行します。

前提条件

- コンテナー化環境で実行中の Red Hat Ceph Storage クラスター
- コンテナーノードへの **root** アクセスがあること。

手順

1. コンテナー名を見つけます。たとえば、**osd.5** に関連付けられたドライブを特定するには、**osd.5** が実行中のコンテナーノードでターミナルを開き、**podman ps** を実行してすべてのコンテナーを一覧表示します。

   例

   ```
   [root@ceph3 ~]# podman ps
   CONTAINER ID  IMAGE                                                COMMAND         CREATED
   STATUS            PORTS             NAMES
   3a866f927b74  registry.redhat.io/rhceph/rhceph-4-rhel8:latest  "/entrypoint.sh"  About
   an hour ago  Up About an hour                ceph-osd-ceph3-sdd
   4e242d932c32  registry.redhat.io/rhceph/rhceph-4-rhel8:latest  "/entrypoint.sh"  About
   an hour ago  Up About an hour                ceph-osd-ceph3-sdc
   91f3d4829079  registry.redhat.io/rhceph/rhceph-4-rhel8:latest  "/entrypoint.sh"  22
   hours ago    Up 22 hours                     ceph-osd-ceph3-sdb
   73dfe4021a49  registry.redhat.io/rhceph/rhceph-4-rhel8:latest  "/entrypoint.sh"  7 days
   ago          Up 7 days                       ceph-osd-ceph3-sdf
   90f6d756af39  registry.redhat.io/rhceph/rhceph-4-rhel8:latest  "/entrypoint.sh"  7 days
   ago          Up 7 days                       ceph-osd-ceph3-sde
   e66d6e33b306  registry.redhat.io/rhceph/rhceph-4-rhel8:latest  "/entrypoint.sh"  7 days
   ago          Up 7 days                       ceph-mgr-ceph3
   733f37aafd23  registry.redhat.io/rhceph/rhceph-4-rhel8:latest  "/entrypoint.sh"  7 days
   ago          Up 7 days                       ceph-mon-ceph3
   ```

2. **podman exec** を使用して、直前の出力から OSD コンテナー名上で **ceph-volume lvm list** を実行します。

   例

   ```
   [root@ceph3 ~]# podman exec ceph-osd-ceph3-sdb ceph-volume lvm list

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

    [journal]    /dev/journals/journal1

       journal uuid              C65n7d-B1gy-cqX3-vZKY-ZoE0-IEYM-HnIJzs
       osd id                    1
   ```

```
    cluster fsid       ce454d91-d748-4751-a318-ff7f7aa18ffd
    type               journal
    osd fsid           661b24f8-e062-482b-8110-826ffe7f13fa
    data uuid          SIEgHe-jX1H-QBQk-Sce0-RUls-8KlY-g8HgcZ
    journal device     /dev/journals/journal1
    data device        /dev/test_group/data-lv2
    devices            /dev/sda

[data]    /dev/test_group/data-lv2

    journal uuid       C65n7d-B1gy-cqX3-vZKY-ZoE0-IEYM-HnIJzs
    osd id             1
    cluster fsid       ce454d91-d748-4751-a318-ff7f7aa18ffd
    type               data
    osd fsid           661b24f8-e062-482b-8110-826ffe7f13fa
    data uuid          SIEgHe-jX1H-QBQk-Sce0-RUls-8KlY-g8HgcZ
    journal device     /dev/journals/journal1
    data device        /dev/test_group/data-lv2
    devices            /dev/sdb
```

この出力から、**osd.5** が **/dev/sdb** に関連付けられていることがわかります。

関連情報

- 詳細は、**「失敗した OSD ディスクの置き換え」**を参照してください。

1.3.3. 同じディスクトポロジーを持つ Ansible を使用した Ceph OSD の追加

同じディスクトポロジーを持つ Ceph OSD の場合、Ansible は **/usr/share/ceph-ansible/group_vars/osds.yml** ファイルの **devices:** セクションで指定されている同じデバイスパスを使用して、他の OSD ノードと同じ OSD 数を追加します。

注記

新しい Ceph OSD ノードは、残りの OSD と同じ設定になります。

前提条件

- 実行中の Red Hat Ceph Storage クラスター
- **『Red Hat Ceph Storage インストールガイド』**の**「Red Hat Ceph Storage のインストール要件」**の章を参照してください。
- 新規ノードへの **root** アクセスがあること。
- ストレージクラスター内の他の OSD ノードと同じ数の OSD データドライブ。

手順

1. **[osds]** セクションの下に Ceph OSD ノードを **/etc/ansible/hosts** ファイルに追加します。

   例

   ```
   [osds]
   ```

```
...
osd06
NEW_OSD_NODE_NAME
```

2. Ansible が Ceph ノードに到達できることを確認します。

```
[user@admin ~]$ ansible all -m ping
```

3. Ansible 設定ディレクトリーに移動します。

```
[user@admin ~]$ cd /usr/share/ceph-ansible
```

4. **ベアメタル** および **コンテナー** のデプロイメントには、Ansible Playbook の **add-osd.yml** を実行します。

```
[user@admin ceph-ansible]$ ansible-playbook infrastructure-playbooks/add-osd.yml -i hosts
```

注記

OSD を追加する際に、**PGs were not reported as active+clean** で Playbook が失敗する場合は、**all.yml** ファイルに以下の変数を設定し、再試行と遅延を調整します。

```
# OSD handler checks
handler_health_osd_check_retries: 50
handler_health_osd_check_delay: 30
```

関連情報

- Ansible インベントリー設定の詳細は、**{storage_product} インストールガイド** の「Ansible のインベントリーの場所の設定」セクションを参照してください。

1.3.4. 異なるディスクトポロジーが設定された Ansible を使用した Ceph OSD の追加

異なるディスクトポロジーを持つ Ceph OSD については、新しい OSD ノードを既存のストレージクラスターに追加する 2 つの方法があります。

前提条件

- 実行中の Red Hat Ceph Storage クラスター
- 『Red Hat Ceph Storage インストールガイド』の「Red Hat Ceph Storage のインストール要件」の章を参照してください。
- 新規ノードへの **root** アクセスがあること。

手順

1. **最初の操作**

   a. **[osds]** セクションの下に、新しい Ceph OSD ノードを **/etc/ansible/hosts** ファイルに追加します。

例

```
[osds]
...
osd06
NEW_OSD_NODE_NAME
```

b. ストレージクラスターに追加される新しい Ceph OSD ノードを **/etc/ansible/host_vars/** ディレクトリーに作成します。

構文

```
touch /etc/ansible/host_vars/NEW_OSD_NODE_NAME
```

例

```
[root@admin ~]# touch /etc/ansible/host_vars/osd07
```

c. 新しいファイルを編集し、**devices:** および **dedicated_devices:** セクションをファイルに追加します。以下の各セクションに、**-** およびスペースを追加してから、この OSD ノードのブロックデバイス名への完全パスを追加します。

例

```
devices:
  - /dev/sdc
  - /dev/sdd
  - /dev/sde
  - /dev/sdf

dedicated_devices:
  - /dev/sda
  - /dev/sda
  - /dev/sdb
  - /dev/sdb
```

d. Ansible がすべての Ceph ノードに到達できることを確認します。

```
[user@admin ~]$ ansible all -m ping
```

e. ディレクトリーを Ansible 設定ディレクトリーに移動します。

```
[user@admin ~]$ cd /usr/share/ceph-ansible
```

f. **ベアメタル** および **コンテナー** のデプロイメントには、Ansible Playbook の **add-osd.yml** を実行します。

```
[user@admin ceph-ansible]$ ansible-playbook infrastructure-playbooks/add-osd.yml -i
hosts
```

2. **2 つ目の方法**

a. 新しい OSD ノード名を **/etc/ansible/hosts** ファイルに追加し、**devices** オプションおよび **dedicated_devices** オプションを使用して、異なるディスクトポロジーを指定します。

例

```
[osds]
...
osd07 devices="['/dev/sdc', '/dev/sdd', '/dev/sde', '/dev/sdf']" dedicated_devices="
['/dev/sda', '/dev/sda', '/dev/sdb', '/dev/sdb']"
```

b. Ansible がすべての Ceph ノードに到達できることを確認します。

```
[user@admin ~]$ ansible all -m ping
```

c. ディレクトリーを Ansible 設定ディレクトリーに移動します。

```
[user@admin ~]$ cd /usr/share/ceph-ansible
```

d. **ベアメタル** および **コンテナー** のデプロイメントには、Ansible Playbook の **add-osd.yml** を実行します。

```
[user@admin ceph-ansible]$ ansible-playbook infrastructure-playbooks/add-osd.yml -i
hosts
```

関連情報

- Ansible インベントリー設定の詳細は、**{storage_product} インストールガイド** の「Ansible のインベントリーの場所の設定」セクションを参照してください。

1.3.5. ceph-volume を使用した Ceph OSD の作成

create サブコマンドは **prepare** サブコマンドを呼び出し、**activate** サブコマンドを呼び出します。

前提条件

- 実行中の Red Hat Ceph Storage クラスター
- Ceph OSD ノードへのルートレベルのアクセス。

注記

作成プロセスに対する制御を強化する場合は、サブコマンドの **prepare** および **activate** を個別に使用して、**create** を使用する代わりに OSD を作成できます。この 2 つのサブコマンドを使用すると、大量のデータをリバランスせずに、新規 OSD をストレージクラスターに段階的に導入することができます。**create** サブコマンドを使用すると、完了直後に OSD が **up** および **in** になりますが、どちらのアプローチも同じように機能します。

手順

1. 新規 OSD を作成するには、以下を実行します。

構文

```
ceph-volume lvm create --bluestore --data VOLUME_GROUP/LOGICAL_VOLUME
```

例

```
[root@osd ~]# ceph-volume lvm create --bluestore --data example_vg/data_lv
```

関連情報

- 詳細は『Red Hat Ceph Storage 管理ガイド』の「`ceph-volume` を使用した Ceph OSD の準備」セクションを参照してください。
- 詳細は、『Red Hat Ceph Storage 管理ガイド』の`ceph-volume` を使用したCeph OSD の有効化セクションを参照してください。

1.3.6. ceph-volumeでのバッチモードの使用

batch サブコマンドは、単一デバイスが提供されると複数の OSD の作成を自動化します。

ceph-volume コマンドは、ドライブタイプに基づいて OSD の作成に使用する最適な方法を決定します。Ceph OSD の最適化は、利用可能なデバイスによって異なります。

- すべてのデバイスが従来のハードドライブの場合、**batch** はデバイスごとに OSD を 1 つ作成します。
- すべてのデバイスがソリッドステートドライブの場合は、**バッチ** によりデバイスごとに OSD が 2 つ作成されます。
- 従来のハードドライブとソリッドステートドライブが混在している場合、**バッチ** はデータに従来のハードドライブを使用し、ソリッドステートドライブに可能な限り大きいジャーナル (**block.db**) を作成します。

注記

batch サブコマンドは、write-ahead-log (**block.wal**) デバイスに別の論理ボリュームを作成することに対応していません。

前提条件

- 実行中の Red Hat Ceph Storage クラスター
- Ceph OSD ノードへのルートレベルのアクセス。

手順

1. 複数のドライブに OSD を作成するには、以下の手順を実行します。

   構文

   ```
   ceph-volume lvm batch --bluestore PATH_TO_DEVICE [PATH_TO_DEVICE]
   ```

   例

   ```
   [root@osd ~]# ceph-volume lvm batch --bluestore /dev/sda /dev/sdb /dev/nvme0n1
   ```

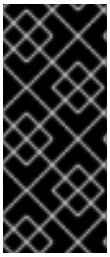
関連情報

- 詳細は、『Red Hat Ceph Storage 管理ガイド』の「`ceph-volume` を使用した Ceph OSD の作成」セクションを参照してください。

1.3.7. コマンドラインインターフェースを使用した Ceph OSD の追加

OSD を Red Hat Ceph Storage に手動で追加するハイレベルのワークフローを以下に示します。

1. **ceph-osd** パッケージをインストールして、新規 OSD インスタンスを作成します。
2. OSD データおよびジャーナルドライブを準備してマウントします。
3. ボリュームグループおよび論理ボリュームを作成します。
4. 新規 OSD ノードを CRUSH マップに追加します。
5. 所有者およびグループパーミッションを更新します。
6. **ceph-osd** デーモンを有効にして起動します。



重要

ceph-disk コマンドは非推奨となりました。**ceph-volume** コマンドは、コマンドラインインターフェースから OSD をデプロイするのに推奨される方法です。現在、**ceph-volume** コマンドは **lvm** プラグインのみをサポートしています。Red Hat は、本ガイドで両方のコマンドを参照として使用している例を提供します。これにより、ストレージ管理者は **ceph-disk** に依存するカスタムスクリプトを **ceph-volume** に変換できます。

注記

カスタムストレージクラスター名の場合は、**ceph** コマンドおよび **ceph-osd** コマンドで **--cluster CLUSTER_NAME** オプションを使用します。

前提条件

- 実行中の Red Hat Ceph Storage クラスター
- 『Red Hat Ceph Storage インストールガイド』の「Red Hat Ceph Storage のインストール要件」の章を参照してください。
- 新規ノードへの **root** アクセス。
- 任意です。**ceph-volume** ユーティリティーでボリュームグループと論理ボリュームを自動的に作成しない場合は、手動で作成します。Red Hat Enterprise Linux 8 の『論理ボリュームの設定および管理』を参照してください。

手順

1. Red Hat Ceph Storage 4 OSD ソフトウェアリポジトリーを有効にします。

   Red Hat Enterprise Linux 7

   ```
   [root@osd ~]# subscription-manager repos --enable=rhel-7-server-rhceph-4-osd-rpms
   ```

Red Hat Enterprise Linux 8

```
[root@osd ~]# subscription-manager repos --enable=rhceph-4-osd-for-rhel-8-x86_64-rpms
```

2. **/etc/ceph/** ディレクトリーを作成します。

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

3. 新しい OSD ノードで、Ceph 管理キーリングと設定ファイルを Ceph Monitor ノードの 1 つからコピーします。

構文

```
scp USER_NAME @ MONITOR_HOST_NAME
:/etc/ceph/CLUSTER_NAME.client.admin.keyring /etc/ceph
scp USER_NAME @ MONITOR_HOST_NAME :/etc/ceph/CLUSTER_NAME.conf /etc/ceph
```

例

```
[root@osd ~]# scp root@node1:/etc/ceph/ceph.client.admin.keyring /etc/ceph/
[root@osd ~]# scp root@node1:/etc/ceph/ceph.conf /etc/ceph/
```

4. **ceph-osd** パッケージを新しい Ceph OSD ノードにインストールします。

Red Hat Enterprise Linux 7

```
[root@osd ~]# yum install ceph-osd
```

Red Hat Enterprise Linux 8

```
[root@osd ~]# dnf install ceph-osd
```

5. OSD を準備します。

- 作成した論理ボリュームを使用するには、以下を実行します。

  構文

  ```
  ceph-volume lvm prepare --bluestore --data VOLUME_GROUP/LOGICAL_VOLUME
  ```

- **ceph-volume** が論理ボリュームを自動的に作成する RAW デバイスを指定するには、以下のコマンドを実行します。

  構文

  ```
  ceph-volume lvm prepare --bluestore --data /PATH_TO_DEVICE
  ```

  詳細は、「OSD の準備」セクションを参照してください。

6. **noup** オプションを設定します。

```
[root@osd ~]# ceph osd set noup
```

7. 新しい OSD をアクティベートします。

構文

```
ceph-volume lvm activate --bluestore OSD_ID OSD_FSID
```

例

```
[root@osd ~]# ceph-volume lvm activate --bluestore 4 6cc43680-4f6e-4feb-92ff-9c7ba204120e
```

詳細は、「OSD のアクティベート」セクションを参照してください。

注記

1 つのコマンドで OSD を準備してアクティベートできます。詳細は、「OSD の作成」セクションを参照してください。または、1 つのコマンドで、複数のドライブを指定し、OSD を作成することもできます。「batch モードの使用」を参照してください。

8. OSD を CRUSH マップに追加します。複数のバケットを指定する場合、コマンドは OSD を指定したバケットから最も具体的なバケットに配置、**および** 指定した他のバケットに従ってバケットを移動します。

構文

```
ceph osd crush add OSD_ID WEIGHT [ BUCKET_TYPE = BUCKET_NAME ...]
```

例

```
[root@osd ~]# ceph osd crush add 4 1 host=node4
```

注記

複数のバケットを指定する場合、コマンドは OSD を指定したバケットから最も具体的なバケットに配置、**および** 指定した他のバケットに従ってバケットを移動します。

注記

CRUSH マップを手動で編集することもできます。**『Red Hat Ceph Storage 戦略ガイド』**の「CRUSH マップの編集」セクションを参照してください。

重要

ルートバケットのみを指定する場合、OSD はルートに直接アタッチしますが、CRUSH ルールは OSD がホストバケット内に置かれることを想定します。

9. **noup** オプションの設定を解除します。

```
[root@osd ~]# ceph osd unset noup
```

10. 新規作成されたディレクトリーの所有者とグループのパーミッションを更新します。

    構文

    ```
    chown -R OWNER : GROUP PATH_TO_DIRECTORY
    ```

    例

    ```
    [root@osd ~]# chown -R ceph:ceph /var/lib/ceph/osd
    [root@osd ~]# chown -R ceph:ceph /var/log/ceph
    [root@osd ~]# chown -R ceph:ceph /var/run/ceph
    [root@osd ~]# chown -R ceph:ceph /etc/ceph
    ```

11. カスタムでストレージクラスターを使用する場合は、以下の行を適切なファイルに追加します。

    ```
    [root@osd ~]# echo "CLUSTER=CLUSTER_NAME" >> /etc/sysconfig/ceph
    ```

    CLUSTER_NAME は、カスタムストレージクラスター名に置き換えます。

12. 新規 OSD が **起動** し、データを受信する準備ができていることを確認するには、OSD サービスを有効にして起動します。

    構文

    ```
    systemctl enable ceph-osd@OSD_ID
    systemctl start ceph-osd@OSD_ID
    ```

    例

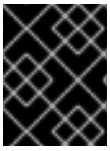
    ```
    [root@osd ~]# systemctl enable ceph-osd@4
    [root@osd ~]# systemctl start ceph-osd@4
    ```

関連情報

- 詳細は、**『Red Hat Ceph Storage 戦略ガイド』**の**「CRUSH マップの編集」**セクションを参照してください。
- **ceph-volume** コマンドの使用に関する詳細は、**『Red Hat Ceph Storage 管理ガイド』**を参照してください。

1.3.8. Ansible を使用した Ceph OSD の削除

Red Hat Ceph Storage クラスターの容量をスケールダウンしないといけない場合があります。Ansible を使用して Red Hat Ceph Storage クラスターから OSD を削除するには、Playbook の **shrink-osd.yml** を実行します。



重要

ストレージクラスターから OSD を削除すると、その OSD に含まれるすべてのデータが破棄されます。

重要

OSD を削除する前に、クラスターにリバランスするのに十分な容量があることを確認します。

重要

配置グループが **active+clean** 状態にあり、OSD に同じオブジェクトのレプリカやイレイジャーコーディングシャードが含まれない限り、OSD を同時に削除しないでください。不明な場合は、Red Hat サポート にお問い合わせください。

前提条件

- Ansible によりデプロイされた実行中の Red Hat Ceph Storage
- 実行中の Ansible 管理ノード

手順

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

   構文

   ```
   [user@admin ~]$ cd /usr/share/ceph-ansible
   ```

2. Ceph Monitor ノードの **/etc/ceph/** から、削除する OSD が含まれるノードに管理キーリングをコピーします。

3. Ceph の通常のデプロイメントまたはコンテナー化されたデプロイメント向けに Ansible Playbook を実行します。

   構文

   ```
   ansible-playbook infrastructure-playbooks/shrink-osd.yml -e osd_to_kill=ID -u
   ANSIBLE_USER_NAME -i hosts
   ```

   以下を置き換えます。

   - **ID** は、OSD ノードの ID に置き換えます。複数の OSD を削除するには、OSD ID をコンマで区切ります。
   - **ANSIBLE_USER_NAME** は、Ansible ユーザーの名前に置き換えてください。

   例

   ```
   [user@admin ceph-ansible]$ ansible-playbook infrastructure-playbooks/shrink-osd.yml -e
   osd_to_kill=1 -u user -i hosts
   ```

4. OSD が正常に削除されていることを確認します。

構文

```
[root@mon ~]# ceph osd tree
```

関連情報

- 『Red Hat Ceph Storage インストールガイド』
- Ansible インベントリー設定の詳細は、**{storage_product} インストールガイド** の「Ansible のインベントリーの場所の設定」セクションを参照してください。

1.3.9. コマンドラインインターフェースを使用した Ceph OSD の削除

ストレージクラスターから OSD を削除するには、以下のステップを実行する必要があります *クラスターマップの更新* 認証キーの削除* OSD の OSD マップからの削除* **ceph.conf** ファイルから OSD を削除します。

OSD ノードに複数のドライブがある場合は、削除する OSD ごとにこの手順を繰り返して、それぞれのドライブについて OSD を削除する必要がある場合があります。

前提条件

- 実行中の Red Hat Ceph Storage クラスター
- 利用可能な OSD が十分になるようにして、ストレージクラスターが **ほぼ完全** な比率にならないようにしてください。
- OSD ノードへのルートレベルのアクセス。

手順

1. OSD サービスを無効にし、停止します。

   構文

   ```
   systemctl disable ceph-osd@OSD_ID
   systemctl stop ceph-osd@OSD_ID
   ```

   例

   ```
   [root@osd ~]# systemctl disable ceph-osd@4
   [root@osd ~]# systemctl stop ceph-osd@4
   ```

   OSD が停止したら、**停止** します。

2. ストレージクラスターから OSD を削除します。

   構文

   ```
   ceph osd out OSD_ID
   ```

   例

```
[root@osd ~]# ceph osd out 4
```

重要

OSD が削除されると、Ceph は再バランス調整を開始し、データをストレージクラスター内の残りの OSD にコピーします。Red Hat は、次の手順に進む前に、ストレージクラスターが **active+clean** になるまで待つことを推奨します。データの移行を確認するには、以下のコマンドを実行します。

構文

```
[root@mon ~]# ceph -w
```

3. CRUSH マップから OSD を削除して、データを受信しないようにします。

   構文

   ```
   ceph osd crush remove OSD_NAME
   ```

   例

   ```
   [root@osd ~]# ceph osd crush remove osd.4
   ```

注記

OSD およびこれを含むバケットを手動で削除するには、CRUSH マップをコンパイルし、デバイス一覧から OSD を削除して、ホストバケットの項目としてデバイスを削除するか、またはホストバケットを削除することもできます。CRUSH マップにあり、ホストを削除するには、マップを再コンパイルしてからこれを設定します。詳細は、『ストレージ戦略ガイド』の「CRUSH マップのデコンパイル」を参照してください。

4. OSD 認証キーを削除します。

   構文

   ```
   ceph auth del osd.OSD_ID
   ```

   例

   ```
   [root@osd ~]# ceph auth del osd.4
   ```

5. OSD を削除します。

   構文

   ```
   ceph osd rm OSD_ID
   ```

   例

```
[root@osd ~]# ceph osd rm 4
```

6. ストレージクラスターの設定ファイルを編集します。このファイルのデフォルト名は **/etc/ceph/ceph.conf** です。ファイルの OSD エントリーが存在する場合は削除します。

   例

   ```
   [osd.4]
   host = _HOST_NAME_
   ```

7. OSD を手動で追加している場合は、**/etc/fstab** ファイルで OSD への参照を削除します。

8. 更新された設定ファイルを、ストレージクラスター内の他のすべてのノードの **/etc/ceph/** ディレクトリーにコピーします。

   構文

   ```
   scp /etc/ceph/CLUSTER_NAME.conf USER_NAME@HOST_NAME:/etc/ceph/
   ```

   例

   ```
   [root@osd ~]# scp /etc/ceph/ceph.conf root@node4:/etc/ceph/
   ```

1.3.10. コマンドラインインターフェースを使用した BlueStore データベースディスクの置き換え

BlueStore OSD のデータベースパーティションが含まれる BlueStore の **block.db** ディスクを置き換える場合、Red Hat は Ansible を使用したすべての OSD の再デプロイのみをサポートします。破損した **block.db** ファイルは、その **block.db** ファイルに含まれるすべての OSD に影響を与えます。

BlueStore **block.db** ディスクを交換する手順として、各デバイスを順番に out とマークし、データがクラスター全体に複製されるのを待って、OSD を交換し、再度 in とマークします。**OSD_ID** を維持し、置き換えられたディスクで新しい **block.db** パーティションで OSD を再作成できます。これは簡単な手順ですが、多くのデータ移行が必要になります。

注記

block.db デバイスに複数の OSD がある場合は、**block.db** デバイスの OSD ごとにこの手順を実行します。**ceph-volume lvm** 一覧を実行して、**block.db** を表示して関係をブロックします。

前提条件

- 実行中の Red Hat Ceph Storage クラスター
- パーティションを持つストレージデバイス。
- すべてのノードへの root レベルのアクセス。

手順

1. モニターノードで現在の Ceph クラスターのステータスを確認します。

```
[root@mon ~]# ceph status
[root@mon ~]# ceph df
```

2. 置き換える障害のある OSD を特定します。

```
[root@mon ~]# ceph osd tree | grep -i down
```

3. OSD ノードで OSD サービスを停止し、無効にします。

構文

```
systemctl disable ceph-osd@OSD_ID
systemctl stop ceph-osd@OSD_ID
```

例

```
[root@osd1 ~]# systemctl stop ceph-osd@1
[root@osd1 ~]# systemctl disable ceph-osd@1
```

4. モニターノードで OSD を **out** に設定します。

構文

```
ceph osd out OSD_ID
```

例

```
[root@mon ~]# ceph osd out 1
```

5. データが OSD から移行されるまで待機します。

構文

```
while ! ceph osd safe-to-destroy OSD_ID ; do sleep 60 ; done
```

例

```
[root@mon ~]# while ! ceph osd safe-to-destroy 1 ; do sleep 60 ; done
```

6. OSD ノードで OSD デーモンを停止します。

構文

```
systemctl kill ceph-osd@OSD_ID
```

例

```
[root@osd1 ~]# systemctl kill ceph-osd@1
```

7. この OSD が使用しているデバイスについて書き留めておいてください。

構文

```
mount | grep /var/lib/ceph/osd/ceph-OSD_ID
```

例

```
[root@osd1 ~]# mount | grep /var/lib/ceph/osd/ceph-1
```

8. OSD ノードの失敗したドライブパスのマウントポイントのマウントを解除します。

構文

```
umount /var/lib/ceph/osd/CLUSTER_NAME-OSD_ID
```

例

```
[root@osd1 ~] #umount /var/lib/ceph/osd/ceph-1
```

9. バックフィルおよびリバランスを回避するために、**noout** および **norebalance** を設定します。

```
[root@mon ~]# ceph osd set noout
[root@mon ~]# ceph osd set norebalance
```

10. 物理ドライブを置き換えます。ノードのハードウェアベンダーのドキュメントを参照してください。新しいドライブを **/dev/** ディレクトリーの下に表示されるように、ドライブパスを書き留めて作業を続行します。

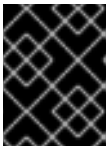
11. monitor ノード上の OSD を破棄します。

構文

```
ceph osd destroy OSD_ID --yes-i-really-mean-it
```

例

```
[root@mon ~]# ceph osd destroy 1 --yes-i-really-mean-it
```



重要

このステップにより、デバイスの内容を破棄します。デバイスのデータが不要であり、クラスターが正常であることを確認します。

12. OSD ディスクの論理ボリュームマネージャーを削除します。

構文

```
lvremove /dev/VOLUME_GROUP/LOGICAL_VOLUME
vgremove VOLUME_GROUP
pvremove /dev/DEVICE
```

例

```
[root@osd1 ~]# lvremove /dev/data-vg1/data-lv1
[root@osd1 ~]# vgremove data-vg1
[root@osd1 ~]# pvremove /dev/sdb
```

13. OSD ノードの OSD ディスクを消去します。

構文

```
ceph-volume lvm zap DEVICE
```

例

```
[root@osd1 ~]# ceph-volume lvm zap /dev/sdb
```

14. OSD ディスクに lvm を再作成します。

構文

```
pvcreate /dev/DEVICE
vgcreate VOLUME_GROUP /dev/DEVICE
lvcreate -l SIZE -n LOGICAL_VOLUME VOLUME_GROUP
```

例

```
[root@osd1 ~]# pvcreate /dev/sdb
[root@osd1 ~]# vgcreate data-vg1 /dev/sdb
[root@osd1 ~]# lvcreate -l 100%FREE -n data-lv1 data-vg1
```

15. 新しい **block.db** ディスクに lvm を作成します。

構文

```
pvcreate /dev/DEVICE
vgcreate VOLUME_GROUP_DATABASE /dev/DEVICE
lvcreate -LI SIZE -n LOGICAL_VOLUME_DATABASE VOLUME_GROUP_DATABASE
```

例

```
[root@osd1 ~]# pvcreate /dev/sdb
[root@osd1 ~]# db-vg1 vgdata /dev/sdb
[root@osd1 ~]# lvcreate -l 100%FREE -n lv-db1 vg-db1
```

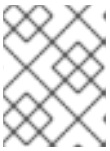
16. OSD ノードで OSD を再作成します。

構文

```
ceph-volume lvm create --bluestore --osd-id OSD_ID --data
VOLUME_GROUP/LOGICAL_VOLUME --block.db
VOLUME_GROUP_DATABASE/LOGICAL_VOLUME_DATABASE
```

例

```
[root@osd1 ~]# ceph-volume lvm create --bluestore --osd-id 1 --data data-vg1/data-lv1 --
block.db db-vg1/db-lv1
```



注記

Red Hat は、以前の手順で破棄されたものと同じ **OSD_ID** を使用することを推奨します。

17. OSD ノードで OSD サービスを起動し、有効にします。

構文

```
systemctl start ceph-osd@OSD_ID
systemctl enable ceph-osd@OSD_ID
```

例

```
[root@osd1 ~]# systemctl start ceph-osd@1
[root@osd1 ~]# systemctl enable ceph-osd@1
```

18. CRUSH 階層をチェックして、OSD がクラスターにあることを確認します。

```
[root@mon ~]# ceph osd tree
```

19. noout および norebalance の設定を解除します。

```
[root@mon ~]# ceph osd unset noout
[root@mon ~]# ceph osd unset norebalance
```

20. **HEALTH_OK** までクラスターステータスをモニターします。

```
[root@mon ~]# watch -n2 ceph -s
```

関連情報

- 詳細は、**Red Hat Ceph Storage インストールガイド** の Red Hat Ceph Storage クラスターのインストールガイド の章を参照してください。

1.3.11. 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。

前提条件

- 実行中の Red Hat Ceph Storage クラスター
- 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。

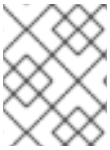
```
[root@monitor ~]# ceph -w
```

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。

1.4. 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。



注記

詳細は、ナレッジベースの記事 How do I increase placement group (PG) count in a Ceph Cluster を参照してください。

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 戦略ガイド』**の**「コードプールの消去」**の章を参照してください。

1.5. 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。

前提条件

- 実行中の Red Hat Ceph Storage クラスター

手順

1. balancer モジュールが有効になっていることを確認します。

```
[root@mon ~]# ceph mgr module enable balancer
```

2. balancer モジュールをオンにします。

```
[root@mon ~]# ceph balancer on
```

3. デフォルトのモードは **crush-compat** です。モードは以下のように変更できます。

```
[root@mon ~]# ceph balancer mode upmap
```

または

```
[root@mon ~]# ceph balancer mode crush-compat
```

ステータス

バランサーの現在のステータスは、以下を実行していつでも確認できます。

```
[root@mon ~]# 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。

```
[root@mon ~]# ceph balancer on
```

以下を使用して、バランサーを再度オフにできます。

```
[root@mon ~]# 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。このパーセンテージは、**max_misplaced** 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。

```
[root@mon ~]# ceph config-key set mgr/balancer/max_misplaced .07
```

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。

     ```
     [root@mon ~]# 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

     ```
     [root@mon ~]# ceph balancer eval rbd
     ```

   - 評価の詳細を表示するには、以下を実行します。

     ```
     [root@mon ~]# 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

     ```
     [root@mon ~]# 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

     ```
     [root@mon ~]# 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

```
[root@mon ~]# 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。

```
[root@mon ~]# 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

```
[root@mon ~]# ceph balancer eval rbd_123
```

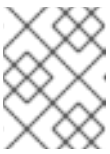
- プランを実行するには、以下を実行します。

構文

```
ceph balancer execute PLAN_NAME
```

例

```
[root@mon ~]# 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、プランは破棄されます。

1.6. 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。**crash dir** オプションを使用して crashdumps を設定できます。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 Monitor 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、それらを **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

```
[root@mon ~]# 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

- 実行中の Red Hat Ceph Storage クラスター

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

```
[root@mon ~]# 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

```
[root@mon ~]# 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

```
[root@mon ~]# 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 を一覧表示します。

例

```
[root@mon ~]# 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 を一覧表示します。

例

```
[root@mon ~]# 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一覧表示します。

例

```
[root@mon ~]# ceph crash stat
8 crashes recorded
8 older than 1 days old:
2021-05-20T08:30:14.533316Z_4ea88673-8db6-4959-a8c6-0eea22d305c2
2021-05-20T08:30:14.590789Z_30a8bb92-2147-4e0f-a58b-a12c2c73d4f5
2021-05-20T08:34:42.278648Z_6a91a778-bce6-4ef3-a3fb-84c4276c8297
2021-05-20T08:34:42.801268Z_e5f25c74-c381-46b1-bee3-63d891f9fc2d
2021-05-20T08:34:42.803141Z_96adfc59-be3a-4a38-9981-e71ad3d55e47
2021-05-20T08:34:42.830416Z_e45ed474-550c-44b3-b9bb-283e3f4cc1fe
2021-05-24T19:58:42.549073Z_b2382865-ea89-4be2-b46f-9a59af7b7a2d
2021-05-24T19:58:44.315282Z_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

```
[root@mon ~]# ceph crash info 2021-05-24T19:58:42.549073Z_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 2021-05-
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": "14.1.0-486.el8cp",
    "crash_id": "2021-05-24T19:58:42.549073Z_b2382865-ea89-4be2-b46f-9a59af7b7a2d",
    "entity_name": "mon.ceph-adm4",
    "os_id": "rhel",
    "os_name": "Red Hat Enterprise Linux",
    "os_version": "8.3 (Ootpa)",
    "os_version_id": "8.3",
    "process_name": "ceph-mon",
    "stack_sig":
"957c21d558d0cba4cee9e8aaf9227b3b1b09738b8a4d2c9f4dc26d9233b0d511",
    "timestamp": "2021-05-24T19:58:42.549073Z",
    "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 Feb 3 03:12:15 EST 2021"
}
```

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

   ```
   [root@mon ~]# 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

   ```
   [root@mon ~]# ceph crash archive 2021-05-24T19:58:42.549073Z_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

```
[root@mon ~]# 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

```
[root@mon ~]# ceph crash rm 2021-05-24T19:58:42.549073Z_b2382865-ea89-4be2-b46f-9a59af7b7a2d
```

関連情報

- Ceph のヘルスメッセージに関する詳細は、**Red Hat Ceph Storage トラブルシューティングガイド** の **Ceph クラスターのヘルスメッセージ** セクションを参照してください。

1.7. 関連情報

- Red Hat Ceph Storage 製品のインストールに関する詳細は、**『Red Hat Ceph Storage インストールガイド』** を参照してください。
- 詳細は、『Red Hat Ceph Storage 戦略ガイド』の**「配置グループ (PG)」**の章を参照してください。
- 詳細は、**『Red Hat Enterprise Linux 8 論理ボリュームの設定および管理』**を参照してください。

第2章 ディスク障害の処理

ストレージ管理者は、ストレージクラスターのライフサイクル時に、特定の時点でディスク障害に対応する必要があります。実際の障害が発生する前にディスク障害をテストおよびシミュレーションすることで、実際に発生したときに備えて準備が整います。

障害の発生したディスクを置き換えるための高度なワークフローを以下に示します。

1. 障害のある OSD を検索します。
2. OSD を取得します。
3. ノード上の OSD デーモンを停止します。
4. Ceph のステータスを確認します。
5. CRUSH マップから OSD を削除します。
6. OSD 認証を削除します。
7. ストレージクラスターから OSD を削除します。
8. ノードのファイルシステムのマウントを解除します。
9. 障害が発生したドライブを置き換えます。
10. OSD をストレージクラスターに追加します。
11. Ceph のステータスを確認します。

2.1. 前提条件

- 実行中の Red Hat Ceph Storage クラスター
- 障害の発生したディスク。

2.2. ディスクの失敗

Ceph は耐障害性を確保できるように設計されているため、Ceph はデータを損失せずに動作が **degraded** の状態になっています。Ceph は、データストレージドライブに障害が発生しても引き続き動作します。**degraded** 状態は、他の OSD に保存されるデータの追加コピーがストレージクラスター内の他の OSD に自動的にバックフィルされることを意味します。OSD が **down** としてマークされる場合は、ドライブに障害が発生したことを意味します。

ドライブに障害が発生すると、最初に OSD ステータスは **down** になりますが、ストレージクラスター内 (**in**) に残ります。ネットワークの問題は、実際に起動 (**up**) していても OSD をダウン (**down**) としてマークすることもできます。まず、環境内のネットワークの問題を確認します。ネットワークが問題がないことを確認する場合は、OSD ドライブが失敗した可能性があります。

最新のサーバーは通常、ホットスワップ可能なドライブでデプロイして、障害が発生したドライブをプルし、ノードを停止せずに新しいドライブに置き換えます。ただし、Ceph では、OSD のソフトウェア定義部分を削除する必要もあります。

2.3. ディスク障害のシミュレーション

ハードとソフト の 2 つのディスク障害シナリオがあります。ハードな障害が発生すると、ディスクが置き換えられます。ソフト障害は、デバイスドライバーまたはその他のソフトウェアコンポーネントに問題がある可能性があります。

ソフトエラーが発生した場合、ディスクの置き換えは不要になる可能性があります。ディスクを置き換える場合、ステップの後に障害の発生したディスクを削除し、交換ディスクを Ceph に追加する必要があります。ソフトディスク障害をシミュレートするために、デバイスを削除するのが最適です。デバイスを選択し、システムからデバイスを削除します。

前提条件

- 正常かつ実行中の Red Hat Ceph Storage クラスター
- Ceph OSD ノードへのルートレベルのアクセス。

手順

1. **sysfs** からブロックデバイスを削除します。

   構文

   ```
   echo 1 > /sys/block/BLOCK_DEVICE/device/delete
   ```

   例

   ```
   [root@osd ~]# echo 1 > /sys/block/sdb/device/delete
   ```

   Ceph OSD のログでは、OSD ノードで Ceph が障害を検出し、復縁プロセスを自動的に開始します。

   例

   ```
   [root@osd ~]# tail -50 /var/log/ceph/ceph-osd.1.log
   2020-09-02 15:50:50.187067 7ff1ce9a8d80  1 bdev(0x563d263d4600 /var/lib/ceph/osd/ceph-2/block) close
   2020-09-02 15:50:50.440398 7ff1ce9a8d80 -1 osd.2 0 OSD:init: unable to mount object store
   2020-09-02 15:50:50.440416 7ff1ce9a8d80 -1 ^[[0;31m ** ERROR: osd init failed: (5) Input/output error^[[0m
   2020-09-02 15:51:10.633738 7f495c44bd80  0 set uid:gid to 167:167 (ceph:ceph)
   2020-09-02 15:51:10.633752 7f495c44bd80  0 ceph version 12.2.12-124.el7cp (e8948288b90d312c206301a9fcf80788fbc3b1f8) luminous (stable), process ceph-osd, pid 36209
   2020-09-02 15:51:10.634703 7f495c44bd80 -1 bluestore(/var/lib/ceph/osd/ceph-2/block) _read_bdev_label failed to read from /var/lib/ceph/osd/ceph-2/block: (5) Input/output error
   2020-09-02 15:51:10.635749 7f495c44bd80 -1 bluestore(/var/lib/ceph/osd/ceph-2/block) _read_bdev_label failed to read from /var/lib/ceph/osd/ceph-2/block: (5) Input/output error
   2020-09-02 15:51:10.636642 7f495c44bd80 -1 bluestore(/var/lib/ceph/osd/ceph-2/block) _read_bdev_label failed to read from /var/lib/ceph/osd/ceph-2/block: (5) Input/output error
   2020-09-02 15:51:10.637535 7f495c44bd80 -1 bluestore(/var/lib/ceph/osd/ceph-2/block) _read_bdev_label failed to read from /var/lib/ceph/osd/ceph-2/block: (5) Input/output error
   2020-09-02 15:51:10.641256 7f495c44bd80  0 pidfile_write: ignore empty --pid-file
   2020-09-02 15:51:10.669317 7f495c44bd80  0 load: jerasure load: lrc load: isa
   2020-09-02 15:51:10.669387 7f495c44bd80  1 bdev create path /var/lib/ceph/osd/ceph-2/block type kernel
   ```

```
2020-09-02 15:51:10.669395 7f495c44bd80  1 bdev(0x55a423da9200
/var/lib/ceph/osd/ceph-2/block) open path /var/lib/ceph/osd/ceph-2/block
2020-09-02 15:51:10.669611 7f495c44bd80  1 bdev(0x55a423da9200
/var/lib/ceph/osd/ceph-2/block) open size 500103643136 (0x7470800000, 466GiB) block_size
4096 (4KiB) rotational
2020-09-02 15:51:10.670320 7f495c44bd80 -1 bluestore(/var/lib/ceph/osd/ceph-2/block)
_read_bdev_label failed to read from /var/lib/ceph/osd/ceph-2/block: (5) Input/output error
2020-09-02 15:51:10.670328 7f495c44bd80  1 bdev(0x55a423da9200
/var/lib/ceph/osd/ceph-2/block) close
2020-09-02 15:51:10.924727 7f495c44bd80  1 bluestore(/var/lib/ceph/osd/ceph-2) _mount
path /var/lib/ceph/osd/ceph-2
2020-09-02 15:51:10.925582 7f495c44bd80 -1 bluestore(/var/lib/ceph/osd/ceph-2/block)
_read_bdev_label failed to read from /var/lib/ceph/osd/ceph-2/block: (5) Input/output error
2020-09-02 15:51:10.925628 7f495c44bd80  1 bdev create path /var/lib/ceph/osd/ceph-
2/block type kernel
2020-09-02 15:51:10.925630 7f495c44bd80  1 bdev(0x55a423da8600
/var/lib/ceph/osd/ceph-2/block) open path /var/lib/ceph/osd/ceph-2/block
2020-09-02 15:51:10.925784 7f495c44bd80  1 bdev(0x55a423da8600
/var/lib/ceph/osd/ceph-2/block) open size 500103643136 (0x7470800000, 466GiB) block_size
4096 (4KiB) rotational
2020-09-02 15:51:10.926549 7f495c44bd80 -1 bluestore(/var/lib/ceph/osd/ceph-2/block)
_read_bdev_label failed to read from /var/lib/ceph/osd/ceph-2/block: (5) Input/output error
```

2. Ceph OSD ディスクツリーを見ると、ディスクがオフラインであると認識されます。

例

```
[root@osd ~]# ceph osd tree
ID WEIGHT  TYPE NAME      UP/DOWN REWEIGHT PRIMARY-AFFINITY
-1 0.28976 root default
-2 0.09659     host ceph3
 1 0.09659         osd.1       down 1.00000          1.00000
-3 0.09659     host ceph1
 2 0.09659         osd.2       up  1.00000          1.00000
-4 0.09659     host ceph2
 0 0.09659         osd.0       up  1.00000          1.00000
```

2.4. 障害のある OSD ディスクの置き換え

OSD を置き換える一般的な手順には、OSD をストレージクラスターから削除し、ドライブを置き換えてから OSD を再作成する必要があります。

前提条件

- 実行中の Red Hat Ceph Storage クラスター
- 障害の発生したディスク。

手順

1. ストレージクラスターの正常性を確認します。

```
[root@mon ~]# ceph health
```

2. CRUSH 階層で OSD の場所を特定します。

```
[root@mon ~]# ceph osd tree | grep -i down
```

3. OSD ノードで、OSD の起動を試行します。

構文

```
systemctl start ceph-osd@OSD_ID
```

コマンドが OSD がすでに実行されていることを示す場合、ハートビートまたはネットワークの問題がある可能性があります。OSD を再起動できない場合は、ドライブが失敗する可能性があります。

注記

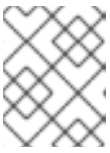
OSD が **down** すると、OSD は最終的に **out** とマークされます。Ceph Storage では、これは通常の動作です。OSD が **out** とマークすると、失敗した OSD のデータのコピーが含まれる他の OSD がバックフィルを開始し、必要な数のコピーがストレージクラスター内に存在していることを確認します。ストレージクラスターがバックフィル状態である間、クラスターの状態は **degraded** になります。

4. Ceph のコンテナー化されたデプロイメントでは、OSD に関連付けられたドライブを参照し、OSD コンテナーを起動します。

構文

```
systemctl start ceph-osd@OSD_DRIVE
```

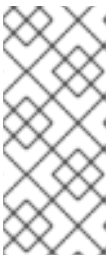
コマンドが OSD がすでに実行されていることを示す場合、ハートビートまたはネットワークの問題がある可能性があります。OSD を再起動できない場合は、ドライブが失敗する可能性があります。



注記

OSD に関連付けられたドライブは、「コンテナー OSD ID をドライブにマッピング」して判断できます。

5. 失敗した OSD のマウントポイントを確認します。



注記

Ceph のコンテナー化されたデプロイメントでは、OSD がダウンし、OSD ドライブのマウントが解除されるため、**df** を実行してマウントポイントを確認することはできません。別の方法を使用して、OSD ドライブが失敗したかどうかを判別します。たとえば、コンテナーノードからドライブで **smartctl** を実行します。

```
[root@osd ~]# df -h
```

OSD を再起動できない場合は、マウントポイントを確認できます。マウントポイントが表示されない場合は、OSD ドライブを再マウントして OSD を再起動することができます。マウントポイントを復元できない場合は、OSD ドライブが失敗している可能性があります。

smartctl ユーティリティー cab を使用して、ドライブが正常かどうかを確認します。

構文

```
yum install smartmontools
smartctl -H /dev/BLOCK_DEVICE
```

例

```
[root@osd ~]# smartctl -H /dev/sda
```

ドライブに障害が発生した場合は、それを置き換える必要があります。

6. OSD プロセスを停止します。

   構文

   ```
   systemctl stop ceph-osd@OSD_ID
   ```

7. Ceph のコンテナー化されたデプロイメントの場合には、OSD に関連付けられたドライブを参照して、OSD コンテナーを停止します。

   構文

   ```
   systemctl stop ceph-osd@OSD_DRIVE
   ```

8. ストレージクラスターから OSD を削除します。

   構文

   ```
   ceph osd out OSD_ID
   ```

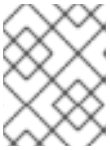
9. 失敗した OSD がバックフィルされていることを確認します。

   ```
   [root@osd ~]# ceph -w
   ```

10. CRUSH マップから OSD を削除します。

    構文

    ```
    ceph osd crush remove osd.OSD_ID
    ```

    注記

    この手順は、OSD を永続的に削除し、再デプロイしない場合にのみ必要になります。

11. OSD の認証キーを削除します。

構文

```
ceph auth del osd.OSD_ID
```

12. OSD のキーが一覧表示されていないことを確認します。

例

```
[root@osd ~]# ceph auth list
```

13. ストレージクラスターから OSD を削除します。

構文

```
ceph osd rm osd.OSD_ID
```

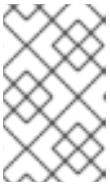
14. 障害が発生したドライブパスのマウントを解除します。

構文

```
umount /var/lib/ceph/osd/CLUSTER_NAME-OSD_ID
```

例

```
[root@osd ~]# umount /var/lib/ceph/osd/ceph-0
```



注記

Ceph のコンテナー化されたデプロイメントでは、OSD がコンテナーがダウンし、OSD ドライブのマウントが解除されます。この場合、マウント解除するものがないため、この手順はスキップできます。

15. 物理ドライブを置き換えます。ノードのハードウェアベンダーのドキュメントを参照してください。ドライブのホットスワップが可能である場合は、障害が発生したドライブを新しいドライブに置き換えます。ドライブがホットスワップできず、ノードに複数の OSD が含まれている場合は、物理ドライブを交換するためにノードを停止する必要がある場合があります。ノードを一時的に停止する必要がある場合は、バックフィルを防ぐためにクラスターを **noout** に設定できます。

例

```
[root@osd ~]# ceph osd set noout
```

ドライブを置き換えて、ノードとその OSD をオンラインに戻したら、**noout** 設定を削除します。

例

```
[root@osd ~]# ceph osd unset noout
```

新しいドライブを **/dev/** ディレクトリーの下に表示されるように、ドライブパスを書き留めて作業を続行します。

16. OSD ドライブを特定し、ディスクをフォーマットします。
17. OSD を再作成します。
    a. 「Ceph Ansible」の使用
    b. 「コマンドラインインターフェース」の使用
18. CRUSH 階層をチェックして、これが正確であることを確認します。

    例

    ```
    [root@osd ~]# ceph osd tree
    ```

    CRUSH 階層の OSD の場所が適切でない場合は、**move** コマンドを使用して移動できます。

    構文

    ```
    ceph osd crush move BUCKET_TO_MOVE BUCKET_TYPE=PARENT_BUCKET
    ```

19. OSD がオンラインであることを確認します。

2.5. OSD ID の保持中に OSD ドライブの置き換え

障害のある OSD ドライブを置き換える場合は、元の OSD ID および CRUSH マップエントリーを保持できます。

注記

ceph-volume lvm コマンドのデフォルトは、OSD 用の BlueStore です。

前提条件

- 実行中の Red Hat Ceph Storage クラスター
- 障害の発生したディスク。

手順

1. OSD を破棄します。

   構文

   ```
   ceph osd destroy OSD_ID --yes-i-really-mean-it
   ```

   例

   ```
   [root@osd ~]# ceph osd destroy 1 --yes-i-really-mean-it
   ```

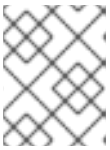
2. 必要に応じて、交換ディスクを以前使用していた場合は、ディスクを **ザッピングする** 必要があります。

   構文

   ```
   ceph-volume lvm zap DEVICE
   ```

   例

   ```
   [root@osd ~]# ceph-volume lvm zap /dev/sdb
   ```

   注記

   DEVICE は、**ceph osd tree**、**ceph osd metadata**、**df** などのさまざまなコマンドの出力を比較することで確認することができます。

3. 既存の OSD ID で新規 OSD を作成します。

   構文

   ```
   ceph-volume lvm create --osd-id OSD_ID --data DEVICE
   ```

   例

   ```
   [root@mon ~]# ceph-volume lvm create --osd-id 1 --data /dev/sdb
   ```

関連情報

- 詳細は、『Red Hat Ceph Storage Operations Guide』の**の「同じディスクトポロジーで Ansible を使用して Ceph OSD 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**の「異なるディスクトポロジーを使用した Ceph OSD 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-volume` を使用した Ceph OSD の準備」**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-volume` を使用した Ceph OSD のアクティベート」**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**「コマンドラインインターフェースを使用した CephOSD の追加」**セクションを参照してください。

第3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。ノードを置き換える際の各シナリオにおけるハイレベルのワークフローを以下に示します。

- 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。
  1. バックフィルを無効にします。
  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。
- 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。
  1. バックフィルを無効にします。
  2. Ceph 設定のバックアップを作成します。
  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。
     a. ディスクを JBOD として設定
  4. オペレーティングシステムをインストールします。
  5. Ceph の設定を復元します。
  6. **ceph-ansible** を実行します。
  7. バックフィルを有効にします。
- 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。
  1. バックフィルを無効にします。
  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。
  3. Ceph 設定のバックアップを作成します。
  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。
     a. ディスクを JBOD として設定
  5. オペレーティングシステムをインストールします。
  6. **ceph-ansible** を実行します。
  7. バックフィルを有効にします。

3.1. 前提条件

- 実行中の Red Hat Ceph Storage クラスター
- 障害のあるノード。

3.2. 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

- Red Hat Ceph Storage ストレージ戦略ガイド

3.3. 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 インターフェー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ースカードはすべて、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。

3.4. 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。

- Ansible の使用
- 「コマンドラインインターフェース」の使用

OSD を追加するには、以下を実行します。

- Ansible の使用
- 「コマンドラインインターフェース」の使用

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、スクラビングの無効化とディープスクラビングの無効化、現在のスクラビングの操作を続行する前に完了させます。

```
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。

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お勧めします。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。

注記

詳細は、ナレッジベースの記事 How do I increase placement group (PG) count in a Ceph Cluster を参照してください。

3.5. 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

- 実行中の Red Hat Ceph Storage クラスター
- 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
- Red Hat Enterprise Linux 8 のインストール
- **『Red Hat Ceph Storage インストールガイド』**の「Red Hat Ceph Storage のインストール要件」の章を参照してください。

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

```
[root@mon ~]# ceph osd set noscrub
[root@mon ~]# 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

```
[root@mon ~]# ceph tell osd.* injectargs --osd-max-backfills 1 --osd-recovery-max-active 1 --osd-recovery-op-priority 1
```

4. 新規ノードを CRUSH マップに追加します。

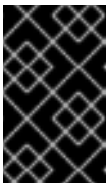
構文

```
ceph osd crush add-bucket BUCKET_NAME BUCKET_TYPE
```

例

```
[root@mon ~]# ceph osd crush add-bucket node2 host
```

5. 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。
   - Ansible の使用
   - 「コマンドラインインターフェース」の使用



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 では、ノード内の一度に 1 つの OSD を追加して、次の OSD に進む前に、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。

関連情報

- 詳細は、**『Red Hat Ceph Storage 設定ガイド』**の**「実行時に特定の構成設定の設定」**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方法は、**『Red Hat Ceph Storage ストレージ戦略ガイド』**の**「バケットの追加」**セクションおよび**「バケットの移動」**セクションを参照してください。

3.6. 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

- 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

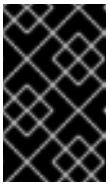
   ```
   ceph tell DAEMON_TYPE.* injectargs --OPTION_NAME VALUE [--OPTION_NAME VALUE]
   ```

   例

   ```
   [root@mon ~]# 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。

   - Ansible の使用
   - 「コマンドラインインターフェース」の使用



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 が削除されると、CRUSH マップからホストバケットを削除します。

構文

```
ceph osd crush rm BUCKET_NAME
```

例

```
[root@mon ~]# ceph osd crush rm node2
```

関連情報

- 詳細は、**『Red Hat Ceph Storage 設定ガイド』**の**「実行時に特定の構成設定の設定」**セクションを参照してください。

3.7. 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

```
[root@ceph1 ~]# ceph df
[root@ceph1 ~]# rados df
[root@ceph1 ~]# 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

   ```
   [root@ceph1 ~]# ceph osd set noout
   [root@ceph1 ~]# ceph osd set noscrub
   [root@ceph1 ~]# 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

   ```
   [root@ceph1 ~]# ceph osd crush rm ceph3
   ```

5. ストレージクラスターのステータスを確認します。

   例

   ```
   [root@ceph1 ~]# 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. Ansible ユーザーを追加して SSH キーを生成します。

   例

   ```
   [root@ceph3 ~]# useradd ansible
   [root@ceph3 ~]# passwd ansible
   [root@ceph3 ~]# cat << EOF > /etc/sudoers.d/ansible
   ansible ALL = (root) NOPASSWD:ALL
   Defaults:ansible !requiretty
   EOF
   [root@ceph3 ~]# su - ansible
   [ansible@ceph3 ~]$ ssh-keygen
   ```

8. Ansible 管理ノードから、再インストールしたノードで **ansible** ユーザーの SSH 鍵をコピーします。

   ```
   [ansible@admin ~]$ ssh-copy-id ceph3
   ```

9. Ansible 管理ノードから Ansible Playbook を再度実行します。

   例

   ```
   [ansible@admin ~]$ cd /usr/share/ceph-ansible
   [ansible@admin ~]$ ansible-playbook site.yml -i hosts

   PLAY RECAP ********************************************************************
   ceph1                      : ok=368  changed=2    unreachable=0    failed=0
   ceph2                      : ok=284  changed=0    unreachable=0    failed=0
   ceph3                      : ok=284  changed=15   unreachable=0    failed=0
   ```

10. 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

```
[root@ceph3 ~]# ceph osd unset noout
[root@ceph3 ~]# ceph osd unset noscrub
[root@ceph3 ~]# ceph osd unset nodeep-scrub
```

11. Ceph のヘルスを確認します。

例

```
[root@ceph3 ~]# ceph -s
    cluster 1e0c9c34-901d-4b46-8001-0d1f93ca5f4d
     health HEALTH_OK
     monmap e1: 3 mons at
{ceph1=192.168.122.81:6789/0,ceph2=192.168.122.82:6789/0,ceph3=192.168.122.83:6789/0}

            election epoch 36, quorum 0,1,2 ceph1,ceph2,ceph3
     osdmap e95: 3 osds: 3 up, 3 in
            flags sortbitwise
      pgmap v1190: 152 pgs, 12 pools, 1024 MB data, 441 objects
            3197 MB used, 293 GB / 296 GB avail
                 152 active+clean
```

関連情報

- 『Red Hat Ceph Storage インストールガイド』
- Ansible インベントリー設定の詳細は、**{storage_product} インストールガイド** の「Ansible のインベントリーの場所の設定」セクションを参照してください。

第4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

4.1. 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。

4.2. 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一覧表示します。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。

4.3. 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構成は、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。全ノードは、階層ツリーの標準 **ルート** であるデフォルトの **ルート** 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

```
[root@mon ~]# ceph osd tree
ID WEIGHT  TYPE NAME          UP/DOWN REWEIGHT PRIMARY-AFFINITY
-1 0.33554 root default
-2 0.04779     host ceph-node3
 0 0.04779         osd.0           up  1.00000          1.00000
-3 0.04779     host ceph-node2
 1 0.04779         osd.1           up  1.00000          1.00000
-4 0.04779     host ceph-node1
 2 0.04779         osd.2           up  1.00000          1.00000
-5 0.04779     host ceph-node4
 3 0.04779         osd.3           up  1.00000          1.00000
-6 0.07219     host ceph-node6
 4 0.07219         osd.4           up  0.79999          1.00000
-7 0.07219     host ceph-node5
 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。

- ノードの ceph-node1 および ceph-node2 がデータセンター 1 (DC1) にある。
- ノードの ceph-node3 および ceph-node5 がデータセンター 2 (DC2) にある。
- ノードの ceph-node4 および ceph-node6 はデータセンター 3 (DC3) にある。
- 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 ceph-node1 datacenter=DC1
ceph osd crush move ceph-node2 datacenter=DC1
ceph osd crush move ceph-node3 datacenter=DC2
ceph osd crush move ceph-node5 datacenter=DC2
ceph osd crush move ceph-node4 datacenter=DC3
ceph osd crush move ceph-node6 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

```
[root@mon ~]# 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 ceph-node1
  2 1.00000             osd.2          up  1.00000          1.00000
 -3 1.00000         host ceph-node2
  1 1.00000             osd.1          up  1.00000          1.00000
-10 2.00000     datacenter DC2
 -2 1.00000         host ceph-node3
  0 1.00000             osd.0          up  1.00000          1.00000
 -7 1.00000         host ceph-node5
  5 1.00000             osd.5          up  0.79999          1.00000
-11 2.00000     datacenter DC3
 -6 1.00000         host ceph-node6
  4 1.00000             osd.4          up  0.79999          1.00000
 -5 1.00000         host ceph-node4
  3 1.00000             osd.3          up  1.00000          1.00000
 -1       0 root default
```

上記の一覧には、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 戦略ガイド』の「CRUSH 管理」の章を参照してください。

第5章 コンテナー化されていない RED HAT CEPH STORAGE クラスターのコンテナー化環境への移行

非コンテナー化のベアメタル、Red Hat Ceph Storage クラスターをコンテナー化環境に手動で移行するには、ceph-ansible **switch-from-non-containerized-to-containerized-ceph-daemons.yml** Playbook を使用します。

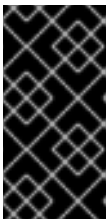
前提条件

- Red Hat Ceph Storage の非コンテナー化、ベアメタル、クラスターの実行
- Ansible 管理ノードへのアクセス
- ansible ユーザーアカウント
- ansible ユーザーアカウントへの sudo アクセス

手順

1. **group_vars/all.yml** ファイルを編集し、コンテナーの設定を追加します。

```
ceph_docker_image_tag: "latest"
ceph_docker_image: rhceph/rhceph-4-rhel8
containerized_deployment: true
```



重要

ceph_docker_image_tag の場合、**現在のストレージクラスターが最新バージョンである場合は** latest を使用するか、適切なイメージタグを使用します。Red Hat Ceph ストレージのリリースと対応する Ceph パッケージのバージョンは何ですか？を参照。

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

```
[ansible@admin ~]$ cd /usr/share/ceph-ansible
```

3. Ansible 管理ノードで、Ansible 移行 Playbook を実行します。

構文

```
ansible-playbook ./infrastructure-playbooks/switch-from-non-containerized-to-containerized-ceph-daemons.yml -i INVENTORY_FILE
```

例

```
[ansible@admin ceph-ansible]$ ansible-playbook ./infrastructure-playbooks/switch-from-non-containerized-to-containerized-ceph-daemons.yml -i hosts
```

クラスターがコンテナー化環境に切り替えていることを確認します。

4. モニターノードで、実行中のコンテナーを一覧表示します。

Red Hat Enterprise Linux 7

```
[root@mon ~]$ sudo docker ps
```

Red Hat Enterprise Linux 8

```
[root@mon ~]$ sudo podman ps
```

関連情報

- ベアメタルストレージクラスターのインストールについての詳細は、**Red Hat Ceph Storage インストールガイド** の Red Hat Ceph Storage クラスターのインストールの章を参照してください。
- Ansible ユーザーに **sudo** アクセスを提供する場合は、**Red Hat Ceph Storage インストールガイド** の sudo アクセスを使用した Ansible ユーザーの作成 セクションを参照してください。