Red Hat Ceph Storage 3

設定ガイド

Red Hat Ceph Storage の設定

Last Updated: 2022-09-06

Red Hat Ceph Storage 3 設定ガイド

Red Hat Ceph Storage の設定

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

法律上の通知

概要

このドキュメントでは、ブート時と実行時に Red Hat Ceph Storage を設定する手順を説明します。また、設定の参考情報も掲載しています。

目次

第1章 設定リファレンス

すべての Ceph クラスターには、以下の項目を定義する設定があります。

- クラスター ID
- 認証設定
- クラスター内の Ceph daemon のメンバーシップ
- ネットワーク設定
- ホスト名およびアドレス
- キーリングへのパス
- データへのパス (ジャーナルを含む)
- 他のランタイムオプション

Red Hat Storage Console や Ansible などのデプロイメントツールは、通常、Ceph の初期設定ファイルを作成します。ただし、デプロイメントツールを使用してクラスターをブートストラップする場合には、独自に作成することができます。

便宜上、各デーモンにはデフォルト値のセットがあります。その多くは **ceph/src/common/config_opts.h** スクリプトで設定されます。これらの設定は、Ceph 設定ファイル、またはランタイム時にモニターの **tell** コマンドを使用するか、Ceph ノード上のデーモンソケットに直接接続して上書きできます。

1.1. 一般的な推奨事項

Ceph 設定ファイルはどこでも維持できますが、Red Hat では、Ceph 設定ファイルのマスターコピーを維持する管理ノードを用意することをお勧めします。

Ceph 設定ファイルに変更を加える場合、一貫性を維持するために、更新された設定ファイルを Ceph ノードにプッシュすることをお勧めします。

1.2. 設定ファイルの構造

Ceph 設定ファイルは、開始時に Ceph デーモンを設定し、デフォルト値をオーバーライドします。Ceph 設定ファイルには、**ini** スタイルの構文を使用します。コメントの前にシャープ記号 (#) またはセミコロン (;) を記入して、コメントを追加できます。以下に例を示します。

```
# <--A number (#) sign precedes a comment.
; A comment may be anything.
# Comments always follow a semi-colon (;) or a pound (#) on each line.
# The end of the line terminates a comment.
# We recommend that you provide comments in your configuration file(s).
```

設定ファイルは、Ceph ストレージクラスター内のすべての Ceph デーモン、または特定タイプのすべての Ceph デーモンを起動時に設定できます。一連のデーモンを設定するには、以下のように設定を受け取るプロセスのセクションに設定を含める必要があります。

[global]

説明

[global] の設定は、Ceph Storage クラスターのすべてのデーモンに影響します。

例

auth supported = cephx

[osd]

説明

[osd] の設定は、Ceph Storage クラスター内のすべての **ceph-osd** デーモンに影響し、**[global]** で同じ設定を上書きします。

例

osd journal size = 1000

[mon]

説明

[mon] の下にある設定は、Ceph Storage クラスター内のすべての **ceph-mon** デーモンに影響し、**[global]** で同じ設定を上書きします。

例

mon host = hostname1,hostname2,hostname3 mon addr = 10.0.0.101:6789

[client]

説明

client の下の設定は、すべての Ceph クライアント (たとえば、マウントされた Ceph ブロックデバイス、Ceph Object Gateway など) に影響します。

例

log file = /var/log/ceph/radosgw.log

グローバル設定は、Ceph ストレージクラスターの全デーモンのすべてのインスタンスに影響します。Ceph Storage クラスター内のすべてのデーモンに共通の値には **global** 設定を使用します。次の方法で、各 **[global]** 設定をオーバーライドできます。

1. 特定のプロセスタイプ (**osd**、**mon** など) の設定を変更する。
2. 特定のプロセスの設定を変更する (例: **osd.1**)。

特定のデーモンでオーバーライドするプロセスを除き、グローバル設定を上書きすると、すべての子プロセスが影響を受けます。

一般的なグローバル設定には、認証のアクティブ化が含まれます。以下に例を示します。

```
[global]
#Enable authentication between hosts within the cluster.
auth_cluster_required = cephx
auth_service_required = cephx
auth_client_required = cephx
```

特定の種類のデーモンに適用される設定を指定できます。特定のインスタンスを指定せずに **[osd]** または **[mon]** で設定を指定すると、設定はすべての OSD またはモニターのデーモンにそれぞれ適用されます。

一般的なデーモン全体の設定には、ジャーナルサイズ、ファイルストア設定などの設定が含まれます。次に例を示します。

```
[osd]
osd_journal_size = 1000
```

デーモンの特定インスタンスの設定を指定できます。タイプとインスタンス ID をピリオド (.) で区切って入力することにより、インスタンスを指定することができます。Ceph OSD デーモンのインスタンス ID は常に数値ですが、Ceph モニターの場合は英数字である場合があります。

```
[osd.1]
# settings affect osd.1 only.

[mon.a]
# settings affect mon.a only.
```

デフォルトの Ceph 設定ファイルの場所を順番に並べると、次のようになります。

1. **$CEPH_CONF** (**$CEPH_CONF** 環境変数に続くパス)
2. **-c path/path** (**-c** コマンドライン引数)
3. **/etc/ceph/ceph.conf**
4. **~/.ceph/config**
5. **./ceph.conf** (現在の作業ディレクトリー内)

一般的な Ceph 設定ファイルには、少なくとも以下の設定があります。

```
[global]
fsid = {cluster-id}
mon_initial_members = {hostname}[, {hostname}]
mon_host = {ip-address}[, {ip-address}]

#All clusters have a front-side public network.
#If you have two NICs, you can configure a back side cluster
#network for OSD object replication, heart beats, backfilling,
#recovery, and so on
public_network = {network}[, {network}]
#cluster_network = {network}[, {network}]

#Clusters require authentication by default.
auth_cluster_required = cephx
auth_service_required = cephx
auth_client_required = cephx

#Choose reasonable numbers for your journals, number of replicas
#and placement groups.
osd_journal_size = {n}
osd_pool_default_size = {n}  # Write an object n times.
osd_pool_default_min_size = {n} # Allow writing n copy in a degraded state.
osd_pool_default_pg_num = {n}
osd_pool_default_pgp_num = {n}

#Choose a reasonable crush leaf type.
```

```
#0 for a 1-node cluster.
#1 for a multi node cluster in a single rack
#2 for a multi node, multi chassis cluster with multiple hosts in a chassis
#3 for a multi node cluster with hosts across racks, and so on
osd_crush_chooseleaf_type = {n}
```

1.3. メタ変数

メタ変数は、Ceph ストレージクラスターの設定を大幅に簡素化します。メタ変数が設定値に設定されると、Ceph はそのメタ変数を具体的な値に展開します。

メタ変数は、Ceph 設定ファイルの **[global]** セクション、**[osd]** セクション、**[mon]** セクション、または **[client]** セクション内で使用すると非常に強力です。しかし、管理用ソケットでも使用可能です。Ceph メタ変数は、Bash のシェル拡張に似ています。

Ceph は以下のメタ変数をサポートしています。

$cluster

説明

Ceph ストレージクラスター名に展開します。同じハードウェアで複数の Ceph ストレージクラスターを実行する場合に便利です。

例

/etc/ceph/$cluster.keyring

デフォルト

ceph

$type

説明

インスタントデーモンのタイプに応じて、**osd** または **mon** のいずれかに展開します。

例

/var/lib/ceph/$type

$id

説明

デーモン識別子に拡張します。**osd.0** の場合、これは **0** になります。

例

/var/lib/ceph/$type/$cluster-$id

$host

説明

インスタントデーモンのホスト名に展開します。

$name

説明

$type.$id まで展開します。

例

/var/run/ceph/$cluster-$name.asok

1.4. CEPH ランタイム設定の表示

ランタイム設定を表示するには、Ceph ノードにログインして以下を実行します。

```
ceph daemon {daemon-type}.{id} config show
```

たとえば、**osd.0** の設定を確認する場合は、**osd.0** を含むノードにログインして次を実行します:

```
ceph daemon osd.0 config show
```

追加のオプションについては、デーモンと **help** を指定します。以下に例を示します。

```
ceph daemon osd.0 help
```

1.5. 実行時に特定の設定設定を取得する

実行時に特定の設定設定を取得するには、Ceph ノードにログインして以下を実行します。

```
ceph daemon {daemon-type}.{id} config get {parameter}
```

たとえば、**osd.0** のパブリックアドレスを取得するには、次を実行します。

```
ceph daemon osd.0 config get public_addr
```

1.6. 実行時の特定の設定設定の設定

ランタイム設定を設定するには、次の 2 つの一般的な方法があります。

- Ceph Monitor のブートストラップ
- 管理ソケットを使用して

tell および **injectargs** コマンドを使用してモニターに接続することにより、Ceph ランタイム設定設定を設定できます。このアプローチを使用するには、変更しようとしているモニターとデーモンが実行されている必要があります。

```
ceph tell {daemon-type}.{daemon id or *} injectargs --{name} {value} [--{name} {value}]
```

{daemon-type} を **osd** または **mon** のいずれかに置き換えます。* を使用して、特定のタイプのすべてのデーモンにランタイム設定を適用するか、特定のデーモンの ID (つまり、その番号または名前) を指定できます。たとえば、**osd.0** という名前の **ceph-osd** デーモンのデバッグロギングを **0/5** に変更するには、以下のコマンドを実行します。

```
ceph tell osd.0 injectargs '--debug-osd 0/5'
```

tell コマンドは複数の引数を取るため、**tell** の各引数は一重引用符で囲み、設定の先頭に 2 つのダッシュを付ける必要があります (**'--{config_opt} {opt-val}' '-{config_opt} {opt-val}'**)。引用符には引数を 1 つしか指定しないため、**daemon** コマンドには引用符は必要ありません。

ceph tell コマンドはモニターを通過します。モニターにバインドできない場合でも、**ceph daemon** を使用して設定を変更したいデーモンのホストにログインすることで、変更を行うことができます。以下に例を示します。

```
sudo ceph osd.0 config set debug_osd 0/5
```

1.7. 一般的な設定リファレンス

一般的な設定は、通常、展開ツールによって自動的に設定されます。

fsid

説明

ファイルシステム ID です。クラスターごとに 1 つになります。

型

UUID

必須

いいえ

デフォルト

該当なし。通常、デプロイメントツールによって生成されます。

admin_socket

説明

Ceph モニターがクォーラムを確立しているかどうかにかかわらず、デーモンの管理コマンドを実行するためのソケット

型

文字列

必須

いいえ

デフォルト

/var/run/ceph/$cluster-$name.asok

pid_file

説明

モニターや OSD が自分の PID を書き込むためのファイル。たとえば、**/var/run/$cluster/$type.$id.pid** は、**ceph** クラスターで実行している id **a** を持つ **mon** の /var/run/ceph/mon.a.pid を作成します。**pid file** は、デーモンが正常に停止すると削除されます。プロセスがデーモン化されていない場合 (つまり、**-f** オプションまたは **-d** オプションで実行)、**pid file** は作成されません。

型

文字列

必須

いいえ

デフォルト

chdir

説明

Ceph デーモンが起動してから変更するディレクトリー。デフォルトの / ディレクトリーが推奨されます。

型

文字列

必須

いいえ

デフォルト

/

max_open_files

説明

これが設定されている場合には、Red Hat Ceph Storage クラスターが起動すると Ceph は OS レベルで **max_open_fds** を設定します (つまりファイル記述子の最大数 #)。これにより、Ceph OSD がファイル記述子を使い果たすのを防ぐことができます。

型

64 ビット整数

必須

いいえ

デフォルト

0

fatal_signal_handlers

説明

設定されていると、SEGV、ABRT、BUS、ILL、FPE、XCPU、XFSZ、SYS シグナルのシグナルハンドラーをインストールして、有用なログメッセージを生成します。

型

ブール値

デフォルト

true

1.8. OSD メモリーターゲット

BlueStore は、**osd_memory_target** 設定オプションを使用して、OSD ヒープメモリーの使用を指定されたターゲットサイズで保持します。

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。デフォルトでは、Anisble は値を 4 GB に設定します。デーモンをデプロイする際に、**/usr/share/ceph-ansible/group_vars/all.yml** ファイルで、バイト単位で示している値を変更できます。

例: **osd_memory_target** を 6000000000 バイトに設定

```
ceph_conf_overrides:
  osd:
    osd_memory_target=6000000000
```

Ceph OSD のメモリーキャッシングは、ブロックデバイスが低速である場合に重要となります (例えば、従来のハードドライブの場合)。キャッシュヒットのメリットがソリッドステートドライブの場合よりもはるかに大きいからです。ただし、ハイパーコンバージドインフラストラクチャー (HCI) や他の

アプリケーションなど、他のサービスと OSD を共存させる場合には、この点を考慮する必要があります。

注記

osd_memory_target の値は、従来のハードドライブデバイス用のデバイスごとに 1 つの OSD、NVMe SSD デバイス用のデバイスごとに 2 つの OSD です。**osds_per_device** は **group_vars/osds.yml** ファイルで定義されます。

その他のリソース

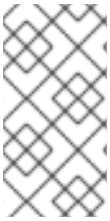
- **osd_memory_target** の 設定 OSD メモリーターゲット の設定

1.9. MDS キャッシュのメモリー制限

MDS サーバーは、そのメタデータを別のストレージプール (**cephfs_metadata**) に保持し、Ceph OSD のユーザーです。Ceph File System の場合、MDS サーバはストレージクラスター内の単一のストレージデバイスだけでなく、Red Hat Ceph Storage クラスター全体をサポートする必要があるため、特にワークロードが小/中サイズのファイルで構成されている場合 (データに対するメタデータの比率が高い)、メモリー要件が大きくなる可能性があります。

例: **mds_cache_memory_limit** を 2000000000 バイトに設定します

```
ceph_conf_overrides:
  osd:
    mds_cache_memory_limit=2000000000
```



注記

メタデータを多用するワークロードを持つ大規模な Red Hat Ceph Storage クラスターでは、MDS サーバーを他のメモリーを多用するサービスと同じノードに置かないでください。そうすることで、より多くのメモリー (たとえば 100 GB を超えるサイズ) を MDS に割り当てることができます。

その他のリソース

- MDS キャッシュサイズの制限 についてを参照してください。

第2章 ネットワーク設定リファレンス

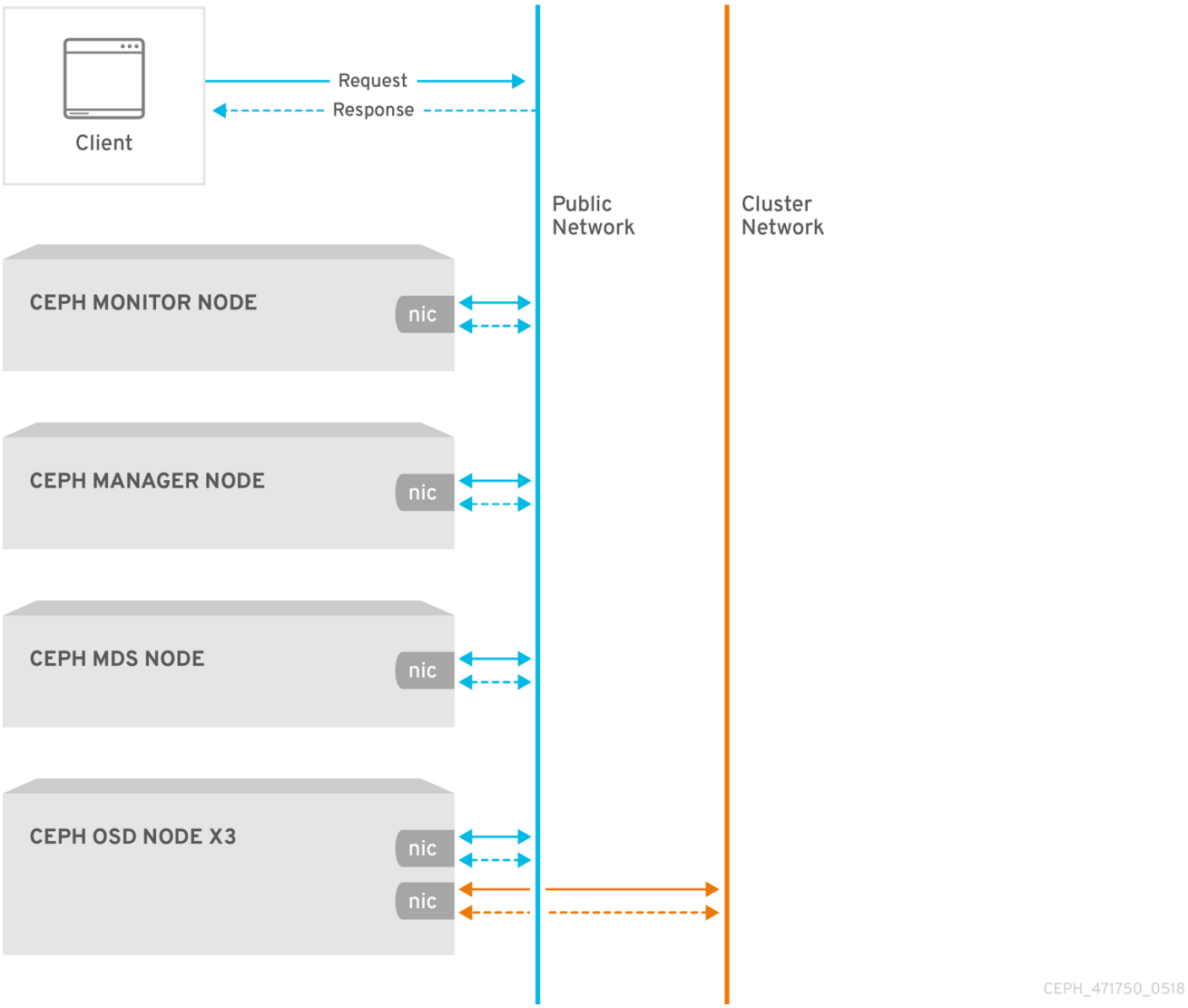
高性能の Red Hat Ceph Storage クラスターを構築するには、ネットワーク設定が重要です。Ceph ストレージクラスターは、Ceph クライアントに代わって要求のルーティングやディスパッチを実行しません。代わりに、Ceph クライアントは Ceph OSD デーモンに直接要求を出します。Ceph OSD は Ceph クライアントに代わってデータレプリケーションを実行するため、レプリケーションおよび他の要素によって Ceph ストレージクラスターのネットワークに追加の負荷がかかります。

すべての Cephク ラスターは、パブリックネットワークを使用する必要があります。ただし、クラスター (内部) ネットワークを指定しない限り、Ceph は単一のパブリックネットワークを想定します。Ceph はパブリックネットワークでのみ機能しますが、大規模なクラスター内に 2 番目のクラスターネットワークを使用すると、パフォーマンスが大幅に向上します。

Red Hat では、Ceph ストレージクラスターを 2 つのネットワークで運用することを推奨しています。

- パブリックネットワーク
- かつクラスターネットワークです。

2 つのネットワークをサポートするには、各 Ceph Node に複数のネットワークインターフェイスカード (NIC) が必要になります。



2 つの別々のネットワークを運用することを検討する理由はいくつかあります。

- **パフォーマンス:** Ceph OSD は Ceph クライアントのデータレプリケーションを処理します。Ceph OSD がデータを複数回複製すると、Ceph OSD 間のネットワーク負荷は、Ceph クライアントと Ceph ストレージクラスター間のネットワーク負荷をすぐに阻害してしまいます。これによりレイテンシーが発生し、パフォーマンスに問題が生じます。リカバリーやリバランシングを行うと、パブリックネットワーク上で大きなレイテンシーが発生します。

- **セキュリティー:** 通常、多くのユーザーはサービス拒否 (DoS) 攻撃と呼ばれる攻撃に関与します。Ceph OSD 間のトラフィックが中断されると、ピアリングが失敗し、配置グループが **active + clean** 状態を反映しなくなり、ユーザーがデータを読み書きできなくなる可能性があります。この種の攻撃に対抗するには、インターネットに直接接続しない、完全に独立したクラスターネットワークを維持することが有効です。

2.1. ネットワーク設定設定

ネットワーク設定の定義は必要ありません。Ceph はパブリックネットワークでのみ機能するので、Ceph デーモンを実行するすべてのホストでパブリックネットワークが設定されている必要があります。しかし、Ceph では、複数の IP ネットワークやサブネットマスクなど、より具体的な条件をパブリックネットワークに設定することができます。また、OSD ハートビート、オブジェクトのレプリケーション、およびリカバリートラフィックを処理するために、別のクラスターネットワークを構築することもできます。

設定で定義する IP アドレスと、ネットワーククライアントがサービスにアクセスする際に使用する公開用の IP アドレスを混同しないようにしてください。通常、内部 IP ネットワークは **192.168.0.0** または **10.0.0.0** です。

ヒント

パブリックネットワークまたはクラスターネットワークに複数の IP アドレスとサブネットマスクを指定する場合、ネットワーク内のサブネットは相互にルーティングできる必要があります。さらに、IP テーブルに各 IP アドレス/サブネットを含め、必要に応じてそれらのポートを開いていることを確認してください。

注記

Ceph はサブネットに CIDR 表記を使用します (例: **10.0.0.0/24**)。

ネットワークの設定が完了したら、クラスターの再起動や各デーモンの再起動を行います。Ceph デーモンは動的にバインドするので、ネットワーク設定を変更してもクラスター全体を一度に再起動する必要はありません。

2.1.1. パブリックネットワーク

パブリックネットワークを設定するには、Ceph 設定ファイルの **[global]** セクションに次のオプションを追加します。

```
[global]
    ...
    public_network = <public-network/netmask>
```

パブリックネットワークの設定では、特にパブリックネットワークの IP アドレスとサブネットを定義することができます。特定のデーモンの **public addr** 設定を使用して、静的 IP アドレスまたは **public network** 設定をオーバーライドできます。

public_network

説明

パブリック (フロントエンド) ネットワークの IP アドレスとネットマスク (例: **192.168.0.0/24**)。**[global]** に設定します。コンマ区切りのサブネットを指定できます。

型

<ip-address>/<netmask> [, <ip-address>/<netmask>]

必須

いいえ

デフォルト

該当なし

public_addr

説明

パブリック (フロントサイド) ネットワークの IP アドレスです。各デーモンのセット。

型

IP アドレス

必須

いいえ

デフォルト

該当なし

2.1.2. クラスターネットワーク

クラスターネットワークを宣言した場合、OSD はハートビート、オブジェクトのレプリケーション、およびリカバリートラフィックをクラスターネットワーク上でルーティングします。これにより、単一のネットワークを使用する場合と比較して、パフォーマンスが向上します。クラスターネットワークを設定するには、Ceph 設定ファイルの **[global]** セクションに次のオプションを追加します。

```
[global]
    ...
    cluster_network = <cluster-network/netmask>
```

セキュリティ強化のためは、クラスターネットワークにはパブリックネットワークやインターネットからアクセスできないようにすることが望ましいです。

クラスターネットワーク設定により、クラスターネットワークを宣言し、特にクラスターネットワークの IP アドレスおよびサブネットを定義できます。特定の OSD デーモンの **cluster addr** 設定を使用して、静的 IP アドレスを割り当てるか、または **cluster network** 設定を上書きすることができます。

cluster_network

説明

クラスターネットワークの IP アドレスとネットマスク (例: **10.0.0.0/24**)。**[global]** に設定します。コンマ区切りのサブネットを指定できます。

型

<ip-address>/<netmask> [, <ip-address>/<netmask>]

必須

いいえ

デフォルト

該当なし

cluster_addr

説明

クラスターネットワークの IP アドレスです。各デーモンのセット。

型

アドレス

必須

いいえ

デフォルト

該当なし

2.1.3. MTU 値の確認および設定

最大伝送単位 (MTU) 値は、リンク層で送信される最大パケットのサイズ (バイト単位) です。MTU のデフォルト値は 1500 バイトです。Red Hat は、Red Hat Ceph Storage クラスターには、MTU 値が 9000 バイトのジャンボフレームを使用することを推奨します。

重要

Red Hat Ceph Storageでは、パブリックネットワークとクラスターネットワークの両方で、通信パスにあるすべてのネットワークデバイスに同じ MTU 値がエンドツーエンドで必要となります。Red Hat Ceph Storage クラスターを実稼働環境で使用する前に、環境内のすべてのノードとネットワーク機器で MTU 値が同じであることを確認します。

注記

ネットワークインターフェースをボンディングする場合には、MTU の値はボンディングされたインターフェースでのみ設定する必要があります。新しい MTU 値は、ボンディングデバイスから下層のネットワークデバイスに伝播します。

前提条件

- ノードへのルートレベルのアクセス。

手順

1. 現在の MTU 値を確認します。

   例

   ```
   [root@mon ~]# ip link list
   1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN mode
   DEFAULT group default qlen 1000
       link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
   2: enp22s0f0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP
   mode DEFAULT group default qlen 1000
   ```

   この例では、ネットワークインターフェースは **enp22s0f0** で、MTU の値は **1500** です。

2. オンラインで MTU 値を **一時的に** 変更するには、以下を実行します。

   構文

   ```
   ip link set dev NET_INTERFACE mtu NEW_MTU_VALUE
   ```

   例

   ```
   [root@mon ~]# ip link set dev enp22s0f0 mtu 9000
   ```

3. MTU 値を **永続的に** 変更するには、以下を行います。

   a. その特定のネットワークインターフェースのネットワーク設定ファイルを編集するために開きます。

      構文

      ```
      vim /etc/sysconfig/network-scripts/ifcfg-NET_INTERFACE
      ```

      例

      ```
      [root@mon ~]# vim /etc/sysconfig/network-scripts/ifcfg-enp22s0f0
      ```

   b. 新しい行で、**MTU=9000** オプションを追加します。

      例

      ```
      NAME="enp22s0f0"
      DEVICE="enp22s0f0"
      MTU=9000 (1)
      ONBOOT=yes
      NETBOOT=yes
      UUID="a8c1f1e5-bd62-48ef-9f29-416a102581b2"
      IPV6INIT=yes
      BOOTPROTO=dhcp
      TYPE=Ethernet
      ```

   c. network サービスを再起動します。

      例

      ```
      [root@mon ~]# systemctl restart network
      ```

その他のリソース

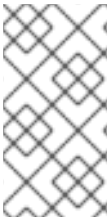
- 詳細は、Red Hat Enterprise Linux 7 の『Networking Guide』を参照してください。

2.1.4. Messaging

メッセンジャーは Ceph ネットワーク層の実装です。Red Hat は 2 種類のメッセンジャーをサポートしています。

- **simple**
- **async**

RHCS 2 以前のリリースでは、**simple** がデフォルトのメッセンジャータイプです。RHCS 3 では、**async** がデフォルトのメッセンジャータイプです。messenger タイプを変更するには、Ceph 設定ファイルの **[global]** セクションに **ms_type** 設定を指定します。



注記

async mesenger では、Red Hat は **posix** トランスポートタイプをサポートしますが、現在 **rdma** または **dpdk** をサポートしていません。デフォルトでは、RHCS 3 の **ms_type** 設定は **async+posix** を反映する必要があります。ここで、**async** はメッセンジャータイプで、**posix** はトランスポートタイプです。

SimpleMessenger について

SimpleMessenger 実装は、1 ソケットあたり 2 つのスレッドを持つ TCP ソケットを使用します。Ceph は、各論理セッションを接続に関連付けます。パイプは、各メッセージの入力と出力を含む接続を処理します。**SimpleMessenger** は、**posix** トランスポートタイプに有効ですが、**rdma**、**dpdk** などの他のトランスポートタイプには有効ではありません。したがって、**AsyncMessenger** は RHCS 3 以降のリリースのデフォルトのメッセンジャータイプです。

AsyncMessenger について

RHCS 3 の場合、**AsyncMessenger** の実装は、接続用の固定サイズのスレッドプールで TCP ソケットを使用します。これは、レプリカまたは消去コードチャンクの最大数と同じにする必要があります。CPU 数が少なかったり、サーバーあたりの OSD 数が多かったりしてパフォーマンスが低下する場合は、スレッドカウントを低い値に設定することができます。

注記

現時点で、Red Hat は **rdma**、**dpdk** などの他のトランスポートタイプをサポートしていません。

メッセンジャーの種類の設定

ms_type

説明

ネットワークトランスポート層のメッセンジャータイプです。Red Hat は、**posix** セマンティクスを使用した、messenger タイプ **simple** および **async** をサポートします。

型

文字列。

必須

いいえ

デフォルト

async+posix

ms_public_type

説明

パブリックネットワークのネットワークトランスポート層のメッセンジャータイプです。これは **ms_type** と同じように動作しますが、パブリックネットワークまたはフロントエンドネットワー

クにのみ適用されます。この設定により、Ceph はパブリックまたはフロントエンドまたはバックサイドのネットワークに異なるメッセンジャータイプを使用できます。

型

文字列。

必須

いいえ

デフォルト

なし。

ms_cluster_type

説明

クラスターネットワークのネットワークトランスポート層のメッセンジャータイプです。これは **ms_type** と同じように動作しますが、クラスターまたはバックサイドネットワークにのみ適用されます。この設定により、Ceph はパブリックまたはフロントエンドまたはバックサイドのネットワークに異なるメッセンジャータイプを使用できます。

型

文字列。

必須

いいえ

デフォルト

なし。

2.1.5. 非同期メッセンジャーの設定

ms_async_transport_type

説明

AsyncMessenger が使用するトランスポートタイプ。Red Hat は **posix** 設定をサポートしますが、現時点では **dpdk** 設定または **rdma** 設定をサポートしません。POSIX は標準的な TCP/IP ネットワークを使用しており、デフォルト値です。その他のトランスポートタイプは実験的なもので、サポートされて **いません**。

型

文字列

必須

いいえ

デフォルト

posix

ms_async_op_threads

説明

各 **AsyncMessenger** インスタンスによって使用されるワーカースレッドの初期数。この設定は、レプリカまたはイレイジャーコードチャンクの数に等しく **なければならない** が、CPU コア数が低い場合や、単一のサーバ上での OSD の数が高い場合には低く設定することもできます。

タイプ

64 ビット未署名の整数

必須

いいえ

デフォルト

3

ms_async_max_op_threads

説明

各 **AsyncMessenger** インスタンスによって使用されるワーカースレッドの最大数。OSD ホストの CPU 数が制限されている場合は低い値に設定し、Ceph が CPU を十分に活用していない場合は高い値に設定します。

型

64 ビット未署名の整数

必須

いいえ

デフォルト

5

ms_async_set_affinity

説明

AsyncMessenger ワーカーを特定の CPU コアにバインドするには、**true** に設定します。

型

ブール値

必須

いいえ

デフォルト

true

ms_async_affinity_cores

説明

ms_async_set_affinity が **true** の場合、この文字列は **AsyncMessenger**ワーカーを CPU コアにバインドする方法を指定します。たとえば、**0,2** はそれぞれワーカー #1 と #2 を CPU コア #0 および #2 にバインドします。**注記:** アフィニティーを手動で設定する場合は、ハイパースレッディングや同様のテクノロジーが原因で作成された仮想 CPU にワーカーを割り当てないようにしてください。これは、物理 CPU コアよりも遅いためです。

型

文字列

必須

いいえ

デフォルト

(empty)

ms_async_send_inline

説明

キューイングや **AsyncMessenger** スレッドから送信せずに、生成したスレッドからメッセージを直接送信します。このオプションは、CPU コア数の多いシステムではパフォーマンスが低下することが知られているため、デフォルトでは無効になっています。

型

ブール値

必須

いいえ

デフォルト

false

2.1.6. バインド

バインド設定は、Ceph OSD デーモンが使用するデフォルトのポート範囲を設定します。デフォルトの範囲は **6800:7100** です。ファイアウォール設定で、設定されたポート範囲を使用できることを確認してください。

また、Ceph デーモンが IPv6 アドレスにバインドするように設定することもできます。

ms_bind_port_min

説明

OSD デーモンがバインドする最小のポート番号。

型

32 ビット整数

デフォルト

6800

必須

▮

ms_bind_port_max

説明

OSD デーモンがバインドする最大のポート番号。

型

32 ビット整数

デフォルト

7300

必須

不要。

ms_bind_ipv6

説明

Ceph デーモンが IPv6 アドレスにバインドするように設定します。

型

ブール値

デフォルト

false

必須

▮

2.1.7. ホスト

Ceph は、Ceph 設定ファイルで少なくとも 1 つのモニターが宣言され、宣言された各モニターの下に **mon addr** が設定されていることを想定しています。Ceph では、Ceph 設定ファイルの宣言されたモニター、メタデータサーバー、および OSD の下に **host** の設定が必要です。

mon_addr

説明

クライアントが Ceph モニターへの接続に使用できる **<hostname>:<port>** エントリーの一覧。設定していない場合には、Ceph は **[mon.*]** セクションを検索します。

型

文字列

必須

いいえ

デフォルト

該当なし

host

説明

ホスト名です。この設定は、特定のデーモンインスタンス (**[osd.0]**など) に使用します。

型

文字列

必須

デーモンインスタンスの場合は Yes。

デフォルト

localhost

ヒント

localhost は使用しないでください。ホスト名を取得するには、**hostname -s** コマンドを実行し、完全修飾ドメイン名ではなく、ホストの名前を最初のピリオドまで使用します。

重要

ホスト名を取得するサードパーティーのデプロイメントシステムを使用する場合は、**host** に値を指定しないでください。

2.1.8. TCP

Ceph はデフォルトで TCP バッファリングを無効にします。

ms_tcp_nodelay

説明

Ceph は **ms_tcp_nodelay** を有効化して、各リクエストが即時に送信されます (バッファーなし)。Nagle アルゴリズムを無効にすると、ネットワークのトラフィックが増加し、混雑の原因となります。小さいパケットが多数ある場合は、**ms_tcp_nodelay** を無効にしてみてください。ただし、通常はこれを無効にすると待ち時間が長くなることに注意してください。

型

ブール値

必須

いいえ

デフォルト

true

ms_tcp_rcvbuf

説明

ネットワーク接続の受信側のソケットバッファのサイズです。デフォルトでは無効になっています。

タイプ

32 ビット整数

必須

いいえ

デフォルト

0

ms_tcp_read_timeout

説明

クライアントまたはデーモンが別の Ceph デーモンへの要求を行い、未使用の接続を解除しない場合、**tcp read timeout** は、指定した秒数後に接続をアイドル状態として定義します。

型

未署名の 64 ビット整数

必須

いいえ

デフォルト

900 15 分。

2.1.9. ファイアウォール

デフォルトでは、デーモンは **6800:7100** 範囲内のポートにバインドされます。この範囲は、ユーザーの判断で設定することができます。ファイアウォールを設定する前に、デフォルトのファイアウォール設定を確認してください。この範囲は、ユーザーの判断で設定することができます。

```
sudo iptables -L
```

firewalld デーモンの場合、**root** として次のコマンドを実行します。

```
# firewall-cmd --list-all-zones
```

一部の Linux ディストリビューションには、すべてのネットワークインターフェースからの SSH を除くすべてのインバウンドリクエストを拒否するルールが含まれています。以下に例を示します。

```
REJECT all -- anywhere anywhere reject-with icmp-host-prohibited
```

2.1.9.1. ファイアウォールの監視

Ceph モニターはデフォルトでポート **6789** をリッスンします。さらに、Ceph モニターは常にパブリックネットワーク上で動作します。以下の例を使用してルールを追加するときは、**<iface>** をパブリックネットワークインターフェイス (たとえば、**eth0**、**eth1** など) に、**<ip-address>** をパブリックネットワークの IP アドレスに、**<netmask>** をパブリックネットワークのネットマスクに置き換えます。

```
sudo iptables -A INPUT -i <iface> -p tcp -s <ip-address>/<netmask> --dport 6789 -j ACCEPT
```

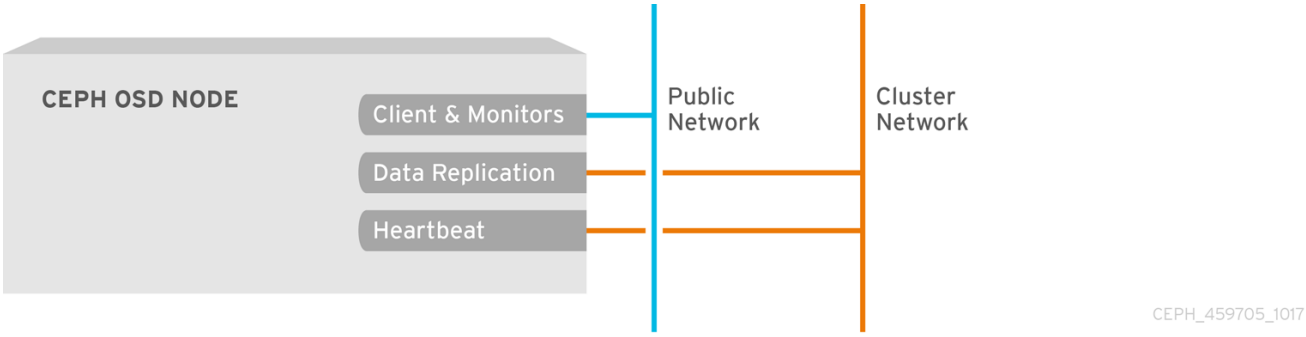
firewalld デーモンの場合、**root** として次のコマンドを実行します。

```
# firewall-cmd --zone=public --add-port=6789/tcp
# firewall-cmd --zone=public --add-port=6789/tcp --permanent
```

2.1.9.2. OSD ファイアウォール

デフォルトでは、Ceph OSD は、ポート 6800 から順に Ceph ノードで最初に利用可能なポートにバインドします。ホストで実行される OSD ごとに、ポート 6800 から始まる少なくとも 3 つのポートを開くようにしてください。

1. クライアントおよびモニターと通信するための 1 つ (パブリックネットワーク)。
2. 1 つは他の OSD (クラスターネットワーク) にデータを送信するためのものです。
3. ハートビートパケットを送信するための 1 つ (クラスターネットワーク)。



ポートはノードごとに異なります。ただし、プロセスが再起動されてバインドされたポートが解放されない場合には、その Ceph ノードで実行されている Ceph デーモンが必要とするポート数よりも多くのポートを開く必要があるかもしれません。デーモンに障害が発生し、ポートを解放せずに再起動した場合に、再起動したデーモンが新しいポートにバインドするように、さらにいくつかのポートを開くことを検討してください。また、各 OSD ホストで **6800:7300** のポート範囲を開くことを検討してください。

パブリックネットワークとクラスターネットワークを別々に設定した場合、クライアントはパブリックネットワークを使用して接続し、他の Ceph OSD デーモンはクラスターネットワークを使用して接続するため、パブリックネットワークとクラスターネットワークの両方にルールを追加する必要があります。

以下の例を使用してルールを追加するときは、**<iface>** をネットワークインターフェイス (**eth0** または **eth1**), `**<ip-address>** を IP アドレスに、**<netmask>** をパブリックまたはクラスターネットワークのネットマスクに置き換えてください。以下に例を示します。

```
sudo iptables -A INPUT -i <iface>  -m multiport -p tcp -s <ip-address>/<netmask> --dports 6800:6810
-j ACCEPT
```

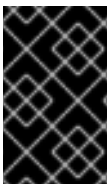
firewalld デーモンの場合、**root** として次のコマンドを実行します。

```
# firewall-cmd --zone=public --add-port=6800-6810/tcp
# firewall-cmd --zone=public --add-port=6800-6810/tcp --permanent
```

クラスターネットワークを別のゾーンに配置した場合は、そのゾーン内のポートを適切に開きます。

2.2. CEPH デーモン

Ceph には、すべてのデーモンに適用される 1 つのネットワーク設定要件があります。Ceph 設定ファイルは、各デーモンに **host** を指定する必要があります。Ceph では、Ceph 設定ファイルでモニター IP アドレスとそのポートを指定する必要がなくなりました。



重要

デプロイメントユーティリティーによっては、設定ファイルを作成してくれる場合もあります。デプロイメントユーティリティーがこれらの値を設定する場合は、設定しないでください。

ヒント

host 設定は、ホストの短縮名です（FQDN ではなく）。IP アドレスでもありません。**hostname -s** コマンドを使用して、ホストの名前を取得します。

```
[mon.a]

    host = <hostname>
    mon addr = <ip-address>:6789

[osd.0]
    host = <hostname>
```

デーモンのホスト IP アドレスを設定する必要はありません。静的 IP 設定があり、パブリックネットワークとクラスターネットワークの両方が実行されている場合、Ceph 設定ファイルは各デーモンのホストの IP アドレスを指定する場合があります。デーモンの静的 IP アドレスを設定するには、Ceph 設定ファイルのデーモンインスタンスセクションに次のオプションを指定する必要があります。

```
[osd.0]
    public_addr = <host-public-ip-address>
    cluster_addr = <host-cluster-ip-address>
```

2 つのネットワーククラスター内に 1 つの NIC OSD

通常、Red Hat は、2 つのネットワークを持つクラスターに単一の NIC を持つ OSD ホストをデプロイすることをお勧めしません。ただし、これを実現するには、Ceph 設定ファイルの **osd.n** セクションに **public addr** エントリーを追加して、OSD ホストを強制的にパブリックネットワーク上で動作させます。**n** は、1 つの NIC を持つ OSD の番号を表します。さらに、パブリックネットワークとクラスターネットワークは互いにトラフィックをルーティングできる必要がありますが、セキュリティ上の理由から Red Hat は推奨していません。

第3章 モニター設定リファレンス

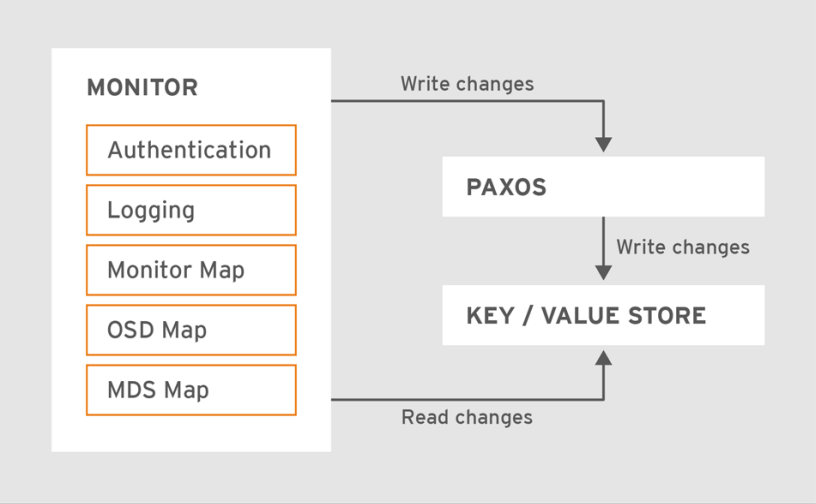
Ceph Monitor の設定方法を理解することは、信頼できる Red Hat Ceph Storage クラスターを構築する上で重要です。すべてのクラスターには少なくとも 1 つのモニターがあります。通常、モニターの設定はほぼ一定のままですが、クラスター内のモニターを追加、削除、または交換することができます。

3.1. 背景情報

Ceph モニターは、クラスターマップの「マスターコピー」を維持します。つまり、1 つの Ceph モニターに接続して最新のクラスターマップを取得するだけで、Ceph クライアントはすべての Ceph モニターと Ceph OSD の位置を把握することができます。

Ceph クライアントが Ceph OSD に対して読み取り/書き込みを行うには、まず Ceph モニターに接続する必要があります。クラスターマップの現在のコピーと CRUSH アルゴリズムを使用して、Ceph クライアントは任意のオブジェクトの位置を計算できます。オブジェクトの位置を計算できることで、Ceph クライアントは Ceph OSD と直接対話できるます。このことは、Ceph の高いスケーラビリティーとパフォーマンスを実現する上で非常に重要な要素となります。

Ceph モニターの主な役割は、クラスターマップのマスターコピーを維持することです。Ceph モニターは、認証とログサービスも提供します。Ceph モニターは、モニターサービスのすべての変更を 1 つの Paxos インスタンスに書き込み、Paxos はその変更をキー/値ストアに書き込んで強い一貫性を持たせます。Ceph モニターは、同期操作中にクラスターマップの最新バージョンにクエリーを行うことができます。Ceph モニターは、(**leveldb** データベースを使用した) キー値ストアのスナップショットやイテレーターを使用して、ストア全体の同期を実行します。



3.1.1. クラスターマップ

クラスターマップは、モニターマップ、OSD マップ、および配置グループマップなどのマップを合成したものです。クラスターマップは、多くの重要なイベントを追跡します。

- どのプロセスが Red Hat Ceph Storage クラスター内 (**in**) にあるか。
- Red Hat Ceph Storage クラスター内 **in** にあるプロセスが **up** で稼働しているか、**down** であるか。
- 配置グループが **active** または **inactive** で **clean** な、または他の一部の状態にあるかどうか。
- クラスターの現状を反映したその他の詳細情報。これには以下が含まれます。
  - ストレージ容量の合計、または

- 使用されているストレージ容量の合計

たとえば、Ceph OSD がダウンしたり、配置グループが低下した状態になったりするなど、クラスターの状態に大きな変化があった場合、クラスターマップはクラスターの現在の状態を反映するように更新されます。さらに、Ceph モニターはクラスターの以前の状態の履歴も保持します。モニターマップ、OSD マップ、および配置グループマップは、それぞれのマップバージョンの履歴を保持します。各バージョンは **エポック** と呼ばれます。

Red Hat Ceph Storage クラスターを操作する場合、これらの状態を追跡することは、クラスター管理の重要な部分です。

3.1.2. クォーラムの監視

クラスターは、1 台のモニターで十分に動作します。しかし、1 台のモニターは単一故障点になります。本番環境の Ceph ストレージクラスターで高可用性を確保するには、複数のモニターで Ceph を実行し、1 つのモニターの故障がクラスター全体の障害にならないようにします。

Ceph ストレージクラスターが高可用性のために複数の Ceph Monitor を実行している場合、Ceph Monitor は Paxos アルゴリズムを使用してマスタークラスターマップに関する合意を確立します。コンセンサスには、クラスターマップに関するコンセンサスのためのクォーラムを確立するために実行されているモニターの過半数が必要です (たとえば、1、3 のうち 2、5 のうち 3、6 のうち 4 など)。

mon_force_quorum_join

詳細

過去にマップから削除されたモニターでも、強制的にクォーラムに参加させます。

型

ブール値

デフォルト

False

3.1.3. 一貫性

Ceph 設定ファイルにモニター設定を追加する場合、Ceph Monitor モニターのアーキテクチャー的な側面をいくつか知っておく必要があります。Cephは、クラスター内で別の Ceph Monitor を検出する際に、Ceph Monitor に厳格な一貫性要件を課します。Ceph クライアントおよびその他の Ceph デーモンは、Ceph 設定ファイルを使用してモニターを検出し、Ceph 設定ファイルではなくモニターマップ (**monmap**) を使用して相互を検出します。

Ceph Monitor が Red Hat Ceph Storage クラスター内の他の Ceph Monitor を検出する場合、常にモニターマップのローカルコピーを参照します。Ceph 設定ファイルの代わりにモニターマップを使用すると、クラスターを壊す可能性のあるエラー (モニターアドレスまたはポートを指定する際の Ceph 設定ファイルの入力ミスなど) を回避できます。モニターは検出のためにモニターマップを使用し、クライアントや他の Ceph デーモンとモニターマップを共有するため、モニターマップは、モニターのコンセンサスが有効であることをモニターに対して厳格に保証します。

厳密な整合性は、モニターマップの更新にも適用されます。Ceph Monitor の他の更新と同様に、モニターマップへの変更は、常に Paxos と呼ばれる分散コンセンサスアルゴリズムを介して実行されます。Ceph Monitor は、モニターマップに対する各更新 (Ceph Monitor の追加や削除など) に同意して、クォーラム内の各モニターが同じバージョンのモニターマップを持つようにする必要があります。モニターマップへの更新はインクリメンタルに行われるため、Ceph Monitor は最新の合意バージョンと以前のバージョンのセットを持つことになります。履歴を維持することで、古いバージョンのモニターマップを持つ Ceph Monitor が、Red Hat Ceph Storage クラスターの現在の状態に追いつくことができます。

Ceph Monitor がモニターマップではなく Ceph 設定ファイルを介してお互いを検出する場合、Ceph 設定ファイルは自動的に更新および配布されないため、新たなリスクが発生する可能性があります。Ceph Monitor が誤って古い Ceph 設定ファイルを使用し、Ceph Monitor の識別に失敗し、クォーラムから外れたり、Paxos がシステムの現在の状態を正確に判断できなかったりする状況が発生する可能性があります。

3.1.4. モニターのブートストラップ

ほとんどの設定およびデプロイメントのケースでは、Ceph をデプロイするツールは、Red Hat Storage Console や Ansible などのモニターマップを生成することで、Ceph Monitor のブートストラップに役立ちます。Ceph モニターには、いくつかの明示的な設定が必要です。

- **ファイルシステム ID**: **fsid** は、オブジェクトストアの一意識別子です。同じハードウェア上で複数のクラスターを稼働させることができるため、モニターのブートストラップを行う場合には、オブジェクトストアの一意の ID を指定する必要があります。たとえば、Red Hat Storage Console または Ansible などのデプロイメントツールを使用すると、ファイルシステム識別子が生成されますが、手動で **fsid** を指定することもできます。

- **モニター ID**: モニター ID は、クラスター内の各モニターに割り当てられる一意の ID です。これは英数字の値であり、慣例により、識別子は通常、アルファベットの増分に従います (たとえば、**a**、**b** など)。これは、Ceph 設定ファイル (**mon.a**、**mon.b** など) で、デプロイメントツールによって、または **ceph** コマンドを使用して設定できます。

- **キー**: モニターには秘密鍵が必要です。

3.2. モニターの設定

クラスター全体に設定を適用するには、**[global]** セクションに構成設定を入力します。クラスター内のすべてのモニターに設定を適用するには、**[mon]** セクションに構成設定を入力します。設定設定を特定のモニターに適用するには、モニターインスタンス (**mon.a** など) を指定します。慣習的に、モニターインスタンス名にはアルファ表記が使用されます。

```
[global]

[mon]

[mon.a]

[mon.b]

[mon.c]
```

3.2.1. 最小設定

Ceph 設定ファイルの Ceph モニターの最低限のモニター設定には、各モニターのホスト名 (DNS に設定されていない場合) とモニターアドレスが含まれます。これらの設定は、**[mon]** の下、または特定のモニターのエントリーの下で設定できます。

```
[mon]
mon_host = hostname1,hostname2,hostname3
mon_addr = 10.0.0.10:6789,10.0.0.11:6789,10.0.0.12:6789
```

または、以下を実行します。

```
[mon.a]
host = hostname1
mon_addr = 10.0.0.10:6789
```

注記

このモニターの最小設定は、デプロイメントツールが **fsid** と **mon.** キーを生成することを前提としています。

重要

Ceph クラスターをデプロイしたら、モニターの IP アドレスを変更しないでください。

RHCS 2.4 の時点では、クラスターが DNS サーバー経由でモニターを検索するように設定されている場合に、Ceph では **mon_host** は必要ありません。DNS ルックアップ用に Ceph クラスターを設定するには、Ceph 設定ファイルの **mon_dns_srv_name** 設定を設定します。

mon_dns_srv_name

説明

モニターのホスト/アドレスを DNS にクエリーする際に使用するサービス名です。

型

文字列

デフォルト

ceph-mon

設定が完了したら、DNS の設定を行います。DNS ゾーンにモニターの IPv4 (A) または IPv6 (AAAA) いずれかのレコードを作成します。以下に例を示します。

```
#IPv4
mon1.example.com. A 192.168.0.1
mon2.example.com. A 192.168.0.2
mon3.example.com. A 192.168.0.3

#IPv6
mon1.example.com. AAAA 2001:db8::100
mon2.example.com. AAAA 2001:db8::200
mon3.example.com. AAAA 2001:db8::300
```

ここで、**example.com** は DNS 検索ドメインになります。

次に、3 つのモニターを指す **mon_dns_srv_name** 設定名で SRV TCP レコードを作成します。以下の例では、デフォルトの **ceph-mon** 値を使用しています。

```
_ceph-mon._tcp.example.com. 60 IN SRV 10 60 6789 mon1.example.com.
_ceph-mon._tcp.example.com. 60 IN SRV 10 60 6789 mon2.example.com.
_ceph-mon._tcp.example.com. 60 IN SRV 10 60 6789 mon3.example.com.
```

モニターはデフォルトでポート **6789** で稼働し、その優先度と重みはすべて前述の例でそれぞれ **10** および **60** に設定されます。

3.2.2. Cluster ID

各 Red Hat Ceph Storage クラスターには固有の ID(**fsid**)があります。指定した場合には、通常は設定ファイルの **[global]** セクションに表示されます。デプロイメントツールは通常、**fsid** を生成してモニターマップに保存するため、値は設定ファイルに表示されない可能性があります。**fsid** を使用すると、同じハードウェア上で複数のクラスターに対してデーモンを実行できます。

fsid

説明

クラスター ID です。クラスターごとに 1 つになります。

型

UUID

必須

Yes

デフォルト

該当なし。指定されていない場合は、デプロイメントツールによって生成されます。

注記

値を設定するデプロイメントツールを使用している場合は、この値を設定しないでください。

3.2.3. 初期メンバー

Red Hat では、高可用性を確保するために、少なくとも 3 つの Ceph Monitor で本番環境の Red Hat Ceph Storage クラスターを実行することを推奨しています。複数のモニターを実行する場合、クォーラムを確立するためにクラスターのメンバーでなければならない初期モニターを指定することができます。これにより、クラスターがオンラインになるまでの時間が短縮される場合があります。

```
[mon]
mon_initial_members = a,b,c
```

mon_initial_members

説明

起動時のクラスター内の最初のモニターの ID です。指定すると、Ceph は最初のクォーラムを形成するための奇数の数のモニターを必要とします (たとえば、3)。

型

文字列

デフォルト

None

注記

クォーラムを確立するには、クラスター内のモニターの **大部分** が相互に到達できる必要があります。この設定でクォーラムを確立するために、モニターの初期数を減らすことができます。

3.2.4. データ

Ceph では、Ceph モニターがデータを保存するデフォルトのパスが用意されています。本番環境の Red Hat Ceph Storage クラスターで最適なパフォーマンスを得るために、Red Hat は、Ceph OSD とは別のホストおよびドライブで Ceph Monitor を実行することを推奨しています。Ceph モニターは **fsync()** 関数を頻繁に呼び出します。これは、Ceph OSD ワークロードに干渉する可能性があります。

Ceph モニターは、データをキー/値ペアとして保存します。データストアを使用すると、他のメリットに加えて、復旧中の Ceph モニターが Paxos と通じて破損したバージョンを実行することを防ぎ、1 つのアトミックバッチで複数の修正操作が可能になります。

注記

Red Hat はデフォルトのデータの場所を変更することを推奨しません。デフォルトの場所を変更する場合は、設定ファイルの **[mon]** セクションにそれを設定して、Ceph モニター全体で統一します。

mon_data

説明

モニターのデータの場所です。

型

文字列

デフォルト

/var/lib/ceph/mon/$cluster-$id

mon_data_size_warn

説明

Ceph は、モニターのデータストアがこのしきい値に達すると、クラスターログで **HEALTH_WARN** ステータスを発行します。デフォルト値は 15GB です。

型

整数

デフォルト

15*1024*1024*1024*`

mon_data_avail_warn

説明

Ceph は、モニターのデータストアで利用可能なディスク領域がこの割合以下になると、クラスターログに **HEALTH_WARN** ステータスを発行します。

型

整数

デフォルト

30

mon_data_avail_crit

説明

Ceph は、モニターのデータストアで利用可能なディスク領域がこの割合以下になると、クラスターログに **HEALTH_ERR** ステータスを発行します。

型

整数

デフォルト

5

mon_warn_on_cache_pools_without_hit_sets

説明

キャッシュプールに **hit_set_type** パラメーターが設定されていないと、Ceph はクラスターログで **HEALTH_WARN** ステータスを発行します。詳細については、プールの値 を参照してください。

タイプ

ブール値

デフォルト

True

mon_warn_on_crush_straw_calc_version_zero

説明

CRUSH の **straw_calc_version** がゼロの場合、Ceph はクラスターログの **HEALTH_WARN** ステータスを発行します。詳細については、CRUSH チューナブル を参照してください。

型

ブール値

デフォルト

True

mon_warn_on_legacy_crush_tunables

説明

CRUSH の調整可能なパラメーターが古くなり過ぎた場合 (**mon_min_crush_required_version** よりも古い場合)、Ceph はクラスターログで **HEALTH_WARN** ステータスを発行します。

型

ブール値

デフォルト

True

mon_crush_min_required_version

説明

この設定では、クラスターが必要とする最小のチューナブルプロファイルバージョンを定義します。詳細については、CRUSH チューナブル を参照してください。

タイプ

文字列

デフォルト

firefly

mon_warn_on_osd_down_out_interval_zero

説明

mon_osd_down_out_interval 設定がゼロの場合、Ceph はクラスターログで **HEALTH_WARN** ステータスを発行します。これは、**noout** フラグが設定されている場合にもリーダーと同様の動作をするためです。管理者は、**noout** フラグを設定してクラスターのトラブルシューティングが容易になります。Ceph は、管理者が設定がゼロであることを認識するために警告を発します。

型

ブール値

デフォルト

True

mon_cache_target_full_warn_ratio

説明

cache_target_full と **target_max_object** の比率で、Ceph により警告が表示されます。

型

浮動小数点 (Float)

デフォルト

0.66

mon_health_data_update_interval

説明

クォーラム内のモニターがピアとヘルスステータスを共有する頻度 (秒単位)。マイナスの数値を入力すると、ヘルスアップデートが無効になります。

型

浮動小数点 (Float)

デフォルト

60

mon_health_to_clog

説明

この設定により、Ceph が定期的にクラスターログにヘルスサマリーを送信することができます。

型

ブール値

デフォルト

True

mon_health_to_clog_tick_interval

説明

モニターが正常性の要約をクラスターログに送信する頻度 (秒単位)。正数以外の数値を指定すると、この設定は無効になります。現在のヘルスサマリーが空であったり、前回と同じであったりする場合、モニターはステータスをクラスターログに送信しません。

型

整数

デフォルト

3600

mon_health_to_clog_interval

説明

モニターが正常性の要約をクラスターログに送信する頻度 (秒単位)。正数以外の数値を指定すると、この設定は無効になります。モニターは常にクラスターログにサマリーを送信します。

型

整数

デフォルト

60

3.2.5. ストレージ容量

Red Hat Ceph Storage クラスターが最大容量 (**mon_osd_full_ratio** パラメーターにより指定) に近くなると、データの損失を防ぐために安全対策として Ceph OSD への書き込みや読み取りができなくなります。そのため、本番環境の Red Hat Ceph Storage クラスターをそのフル比率に近づけてしまうことは、高可用性が犠牲になってしまうのでグッドプラクテイスとは言えません。デフォルトのフル比率は、**.95** (容量の 95%) です。これは、OSD の数が少ないテストクラスター用の非常に厳しい設定です。

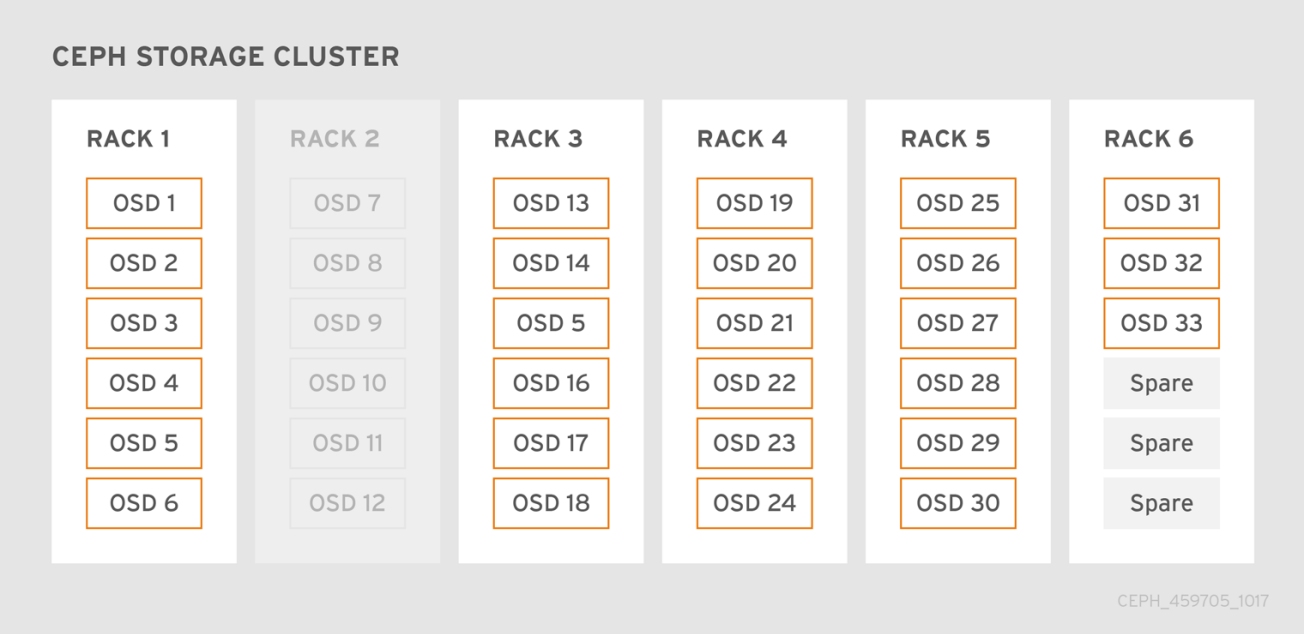
ヒント

クラスターをモニタリングする際に、**nearfull** な比率に関連する警告にアラートしてください。つまり、1 つまたは複数の OSD が故障した場合、一部の OSD の障害により一時的にサービスが中断される可能性があります。ストレージの容量を増やすために、OSD の増設を検討してください。

テストクラスターの一般的なシナリオでは、システム管理者がRed Hat Ceph Storage クラスターから Ceph OSD を削除してクラスターの再バランスを観察します。その後、別の Ceph OSD を削除し、Red Hat Ceph Storage クラスターが最終的にフル比率に達してロックアップするまでこれを繰り返します。

Red Hat では、テストクラスターであっても、多少の容量計画を立てることを推奨しています。計画を立てることで、高可用性を維持するためにどれだけの予備容量が必要なのかを把握することができます。理想的には、Ceph OSD を直ちに置き換えることなく、クラスターが **active + clean** な状態に復元できる Ceph OSD の一連の障害を計画する必要があります。クラスターを **active + degraded** の状態で実行できますが、これは通常の動作条件には理想的ではありません。

次の図は、33 台の Ceph Node が含まれる単純化した Red Hat Ceph Storage クラスターを示しています。ホストごとに 1 つの Ceph OSD があり、各 Ceph OSD デーモンは 3 TB のドライブに対して読み取りおよび書き込を行います。つまり、この例の Red Hat Ceph Storage クラスターの最大実容量は 99 TB です。**mon osd full ratio** が **0.95** の場合は、Red Hat Ceph Storage クラスターが空き容量が 5 TB になると、Ceph クライアントはデータの読み取りと書き込みを許可しません。そのため、Red Hat Ceph Storage クラスターの運用上の容量は 99 TB ではなく 95 TB となります。



このようなクラスターでは、1 つまたは 2 つの OSD が故障するのが普通です。頻度は低いが妥当なシナリオとしては、ラックのルーターや電源が故障し、複数の OSD が同時にダウンすることが挙げられます (例: OSD 7 - 12)。このようなシナリオでは、さらに OSD のあるホストを短い順序で追加する場合でも、動作し続け、**active + clean** な状態を実現するクラスターを試す必要があります。容量利用率が高すぎると、データを失うことはないかもしれませんが、クラスターの容量利用率がフル比率を超えた場合、障害ドメイン内の障害を解決している間データの可用性が犠牲になる可能性があります。このため、Red Hat では、少なくとも大まかな容量計画を立てることを推奨しています。

クラスターに関する 2 つの数字を把握します。

- OSD の数
- クラスターの総容量

クラスター内の OSD の平均容量を求めるには、クラスターの総容量をクラスター内の OSD の数で割ります。この数に、通常の運用で同時に故障すると予想される OSD の数 (比較的小さい数) を乗じます。最後に、クラスターの容量にフル比率を掛けて、運用上の最大容量を算出します。そして、失敗すると予想される OSD からデータ量を差し引いて、合理的なフル比率を算出します。前述のプロセスを、より多くの OSD 故障数 (例えば、OSD のラック) で繰り返し、ほぼフル比率のための妥当な数を算出します。

```
[global]
...
mon_osd_full_ratio = .80
mon_osd_nearfull_ratio = .70
```

mon_osd_full_ratio

説明

OSD が **full** とみなされるまでのディスク領域のパーセンテージ。

型

浮動小数点

デフォルト

.95

mon_osd_nearfull_ratio

説明

OSD がほぼ **nearfull** とみなされるまでのディスク領域のパーセンテージ。

型

浮動小数点 (Float)

デフォルト

.85

ヒント

一部の OSD が **nearfull** で、他の OSD には十分な容量がある場合、 **nearfull** OSD の CRUSH 重みに問題がある可能性があります。

3.2.6. ハートビート

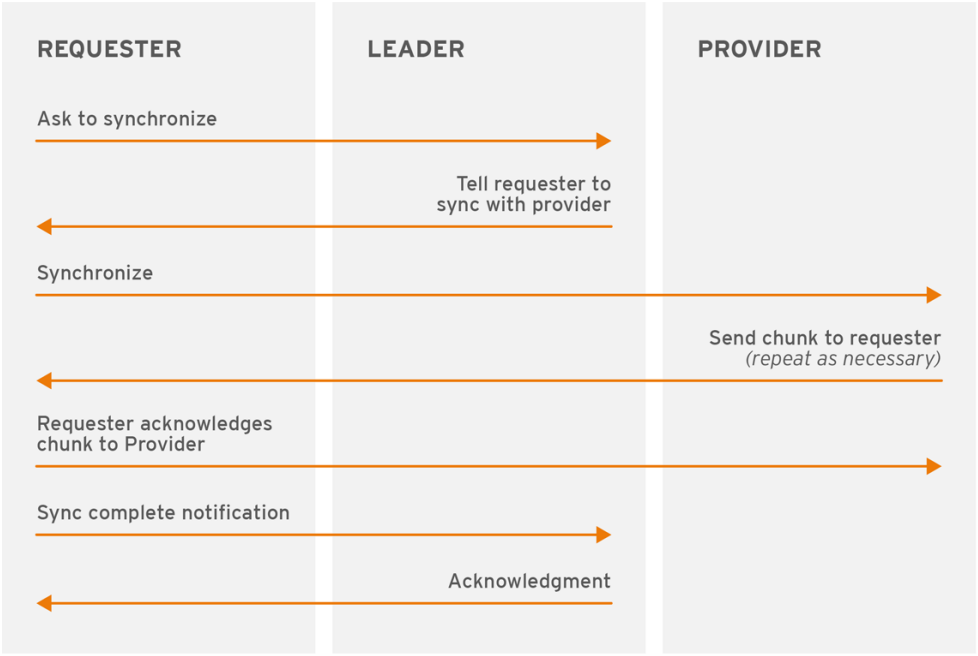
Ceph モニターは、各 OSD からのレポートを要求し、隣接する OSD の状態に関するレポートを OSD から受け取ることで、クラスターについて把握します。Ceph では、モニターと OSD の間の相互作用について妥当なデフォルト設定が用意されていますが、必要に応じて変更することができます。

3.2.7. ストア同期の監視

複数のモニターを持つ本番環境用のクラスターを運用する場合 (推奨される構成)、各モニターは隣接するモニターがより新しいバージョンのクラスターマップを持っているかどうかを確認します。例えば、隣接するモニターのマップのエポックナンバーが、インスタントモニターのマップの最新のエポックより 1 つ以上高い場合。定期的に、クラスター内のあるモニターが他のモニターから遅れをとることがあります。その場合、そのモニターはクォーラムから離脱し、同期をとってクラスターに関する最新の情報を取得した後、再びクォーラムに参加しなければなりません。同期のために、モニターは以下の 3 つのロールのいずれかを取ります。

- **リーダー**: リーダーは、クラスターマップの最新の Paxos バージョンを実現する最初のモニターです。
- **プロバイダー**: プロバイダーは最新バージョンのクラスターマップを持つモニターですが、最新バージョンを最初に達成したわけではありません。
- **リクエスター:** リクエスターはリーダーの背後に置かれたモニターで、クォーラムに再参加する前にクラスターに関する最新情報を取得するために同期する必要があります。

これらのロールにより、リーダーは同期のタスクをプロバイダーに委譲しすることができ、同期の要求によりリーダーが過負荷になることを防ぎ、パフォーマンスが向上します。次の図では、リクエスターが他のモニターに遅れをとっていることを認識しています。リクエスターはリーダーに同期を依頼し、リーダーはリクエスターにプロバイダーとの同期を指示します。



新しいモニターがクラスターに参加すると、常に同期が行われます。実行時の運用において、モニターは異なるタイミングでクラスターマップへの更新を受け取る場合があります。つまり、リーダーとプロバイダーのロールが、モニター間で移動する可能性があるということです。例えば、同期中にこれが起こると、プロバイダはリーダーから遅れてしまい、プロバイダはリクエスターとの同期を終了することができます。

同期が完了すると、Ceph ではクラスター全体のトリミングが必要になります。トリミングを行うには、配置グループが **active + clean** である必要があります。

mon_sync_trim_timeout

説明, 型

double

デフォルト

30.0

mon_sync_heartbeat_timeout

説明, 型

double

デフォルト

30.0

mon_sync_heartbeat_interval

説明, 型

double

デフォルト

5.0

mon_sync_backoff_timeout

説明, 型

double

デフォルト

30.0

mon_sync_timeout

説明

モニターが、更新メッセージをあきらめて再びブートストラップを行うまで、同期プロバイダから次のメッセージを待つ秒数。

型

double

デフォルト

30.0

mon_sync_max_retries

説明, 型

整数

デフォルト

5

mon_sync_max_payload_size

説明

同期ペイロードの最大サイズ (単位: バイト) です。

型

32 ビット整数

デフォルト

1045676

paxos_max_join_drift

説明

モニターデータストアを最初に同期させるまでの、Paxos 最大反復回数です。モニターは、ピアが自分よりも先に進んでいると判断すると、先に進む前にまずデータストアと同期します。

型

整数

デフォルト

10

paxos_stash_full_interval

説明

PaxosService の状態のフルコピーを隠す頻度 (コミット数)。現在、この設定は **mds**、**mon**、**auth**、および **mgr** PaxosServices のみに影響します。

型

整数

デフォルト

25

paxos_propose_interval

説明

この時間間隔で更新情報を集めてから、マップの更新を提案します。

型

double

デフォルト

1.0

paxos_min

説明

維持する paxos の状態の最小数

型

整数

デフォルト

500

paxos_min_wait

説明

活動していない期間の後にアップデートを収集するための最小時間。

型

double

デフォルト

0.05

paxos_trim_min

説明

トリミング前に許容される追加提案の数

型

整数

デフォルト

250

paxos_trim_max

説明

一度にトリミングする追加提案の最大数

型

整数

デフォルト

500

paxos_service_trim_min

説明

トリムのトリガーとなる最小のバージョン数 (0であれば無効)

型

整数

デフォルト

250

paxos_service_trim_max

説明

1回の提案中にトリミングするバージョン数の最大値 (0 であれば無効)

型

整数

デフォルト

500

mon_max_log_epochs

説明

1回の提案中にトリミングするログエポック数の最大値

型

整数

デフォルト

500

mon_max_pgmap_epochs

説明

1回の提案中にトリミングする pgmap エポック数の最大値

型

整数

デフォルト

500

mon_mds_force_trim_to

説明

モニターがこのポイントまで mdsmaps をトリミングするのを強制します (0 は無効、危険なので使用には注意が必要)。

型

整数

デフォルト

0

mon_osd_force_trim_to

説明

指定したエポックでクリーンではない PG があっても、モニターがこのポイントまで osdmaps をトリミングするのを強制します (0 は無効、危険なので使用には注意が必要)。

型

整数

デフォルト

0

mon_osd_cache_size

説明

基礎となるストアのキャッシュに依存しない、osdmaps のキャッシュサイズ

型

整数

デフォルト

10

mon_election_timeout

説明

選択の提案側で、すべての ACK を待つ最長の時間 (秒単位)

型

浮動小数点 (Float)

デフォルト

5

mon_lease

説明

モニターのバージョンのリース期間 (秒単位)

型

浮動小数点 (Float)

デフォルト

5

mon_lease_renew_interval_factor

説明

mon lease * **mon lease renew interval factor** は、リーダーが他のモニターのリースを更新する間隔になります。係数は **1.0** 未満でなければなりません。

型

浮動小数点 (Float)

デフォルト

0.6

mon_lease_ack_timeout_factor

説明

リーダーは、プロバイダーがリース拡張を承認するまで **mon lease** * **mon lease ack timeout factor** を待機します。

型

浮動小数点 (Float)

デフォルト

2.0

mon_accept_timeout_factor

説明

Leader は **mon lease** * **mon accept timeout factor** を待ち、リクエスターが Paxos の更新を受け入れるのを待機します。また、Paxos の回復期にも同様の目的で使用されます。

型

浮動小数点 (Float)

デフォルト

2.0

mon_min_osdmap_epochs

説明

常時保持する OSD マップエポックの最小数

型

32 ビット整数

デフォルト

500

mon_max_pgmap_epochs

説明

モニターが保持すべき PG マップエポックの最大数

型

32 ビット整数

デフォルト

500

mon_max_log_epochs

説明

モニターが保持すべきログエポックの最大数

型

32 ビット整数

デフォルト

500

3.2.8. クロック

Ceph デーモンは、クリティカルなメッセージを相互に渡します。このメッセージは、デーモンがタイムアウトのしきい値に達する前に処理する必要があります。Ceph モニターのクロックが同期していないと、さまざまな異常が発生する可能性があります。以下に例を示します。

- 受信したメッセージを無視するデーモン (タイムスタンプが古いなど)。
- メッセージ受信のタイミングが適切でない場合、タイムアウトの発生が早すぎたり遅すぎたりする。

詳細については、ストア同期の監視 を参照してください。

ヒント

Ceph モニターホストに NTP をインストールして、モニタークラスターのクロックが同期した状態で動作するようにします。

NTP では、遅れによる悪影響が出ていなくても、クロックドリフトが目立つことがあります。NTP が適切なレベルの同期を維持していても、Ceph のクロックドリフトとクロックスキューの警告が発生することがあります。このような状況では、クロックドリフトを増やすことが許容できるかもしれません。ただし、ワークロード、ネットワーク遅延、デフォルトタイムアウトに対するオーバーライドの設定、Monitor Store Synchronization 設定などの多くの要因が、Paxos の保証を損なうことなく、許容可能なクロックドリフトのレベルに影響を与える可能性があります。

Ceph は、受け入れ可能な値を見つけることができるように、次の調整可能なオプションを提供します。

clock_offset

説明

システムクロックをどれだけオフセットするか。詳細は、**Clock.cc** を参照してください。

型

double

デフォルト

0

mon_tick_interval

説明

モニターの目盛りの間隔 (秒単位)

型

32 ビット整数

デフォルト

5

mon_clock_drift_allowed

説明

モニター間で許容されるクロックドリフト (秒単位)

型

浮動小数点 (Float)

デフォルト

.050

mon_clock_drift_warn_backoff

説明

クロックドリフト警告のための指数バックオフ

型

浮動小数点 (Float)

デフォルト

5

mon_timecheck_interval

説明

リーダーの時刻チェック (クロックドリフトチェック) 間隔 (秒単位)

型

浮動小数点 (Float)

デフォルト

300.0

mon_timecheck_skew_interval

説明

スキューがあった場合のリーダーの時刻チェック (クロックドリフトチェック) 間隔 (秒単位)

型

浮動小数点 (Float)

デフォルト

30.0

3.2.9. クライアント

mon_client_hunt_interval

説明

クライアントは、接続が確立されるまで、**N** 秒ごとに新しいモニターを試行します。

タイプ

double

デフォルト

3.0

mon_client_ping_interval

説明

クライアントは、**N** 秒ごとにモニターに ping を送信します。

タイプ

double

デフォルト

10.0

mon_client_max_log_entries_per_message

説明

モニターがクライアントメッセージごとに生成するログエントリーの最大数。

タイプ

整数

デフォルト

1000

mon_client_bytes

説明

メモリー内で許容されるクライアントメッセージデータの量 (バイト単位)。

タイプ

64 ビット整数未署名

デフォルト

100ul << 20

3.3. その他

mon_max_osd

説明

クラスターで許容される OSD の最大数

型

32 ビット整数

デフォルト

10000

mon_globalid_prealloc

説明

クラスター内のクライアントおよびデーモンに事前に割り当てるグローバル ID の数

型

32 ビット整数

デフォルト

100

mon_sync_fs_threshold

説明

指定された数のオブジェクトを書き込む際に、ファイルシステムと同期します。無効にするには **0** に設定します。

型

32 ビット整数

デフォルト

5

mon_subscribe_interval

説明

サブスクリプションの更新間隔 (秒単位)。サブスクリプションメカニズムにより、クラスターマップやログ情報を取得することができます。

型

double

デフォルト

300

mon_stat_smooth_intervals

説明

最後の **N** PG マップに対する統計は、Ceph によりスムーズになります。

型

整数

デフォルト

2

mon_probe_timeout

説明

モニターがブートストラップを行うまで、ピアを探すために待機する秒数

型

double

デフォルト

2.0

mon_daemon_bytes

説明

メタデータサーバーおよび OSD メッセージのメッセージメモリー容量 (単位: バイト)

型

64 ビット整数未署名

デフォルト

400ul << 20

mon_max_log_entries_per_event

説明

1 イベントあたりのログエントリーの最大数

型

整数

デフォルト

4096

mon_osd_prime_pg_temp

説明

クラスター外の OSD がクラスターに戻ってきたときに、以前の OSD で PGMap のプライミングを行うことを有効または無効にします。**true** 設定では、クライアントは、PG のピア化として OSD で新たに実行するまで、以前の OSD を引き続き使用します。

型

ブール値

デフォルト

true

mon_osd_prime_pg_temp_max_time

説明

クラスター外の OSD がクラスターに戻ってきたときに、モニターが PGMAP のプライミングを試みる時間 (秒単位)

型

浮動小数点 (Float)

デフォルト

0.5

mon_osd_prime_pg_temp_max_time_estimate

説明

すべての PG を並行してプライミングするまでに、各 PG での時間の最大推定値

型

浮動小数点 (Float)

デフォルト

0.25

mon_osd_allow_primary_affinity

説明

osdmap で **primary_affinity** を設定できるようにします。

型

ブール値

デフォルト

False

mon_osd_pool_ec_fast_read

説明

プールでの高速読み込みオンにするかどうか。作成時に **fast_read** が指定されていない場合に、新たに作成されたイレイジャープールのデフォルト設定として使用します。

型

ブール値

デフォルト

False

mon_mds_skip_sanity

説明

バグ発生に関わらず続行したい際に、FSMap の安全アサーションをスキップします。FSMap のサニティーチェックに失敗すると Monitor は終了しますが、このオプションを有効にすることでそれを無効にすることができます。

型

ブール値

デフォルト

False

mon_max_mdsmap_epochs

説明

1回の提案中にトリミングする mdsmap エポック数の最大値

型

整数

デフォルト

500

mon_config_key_max_entry_size

説明

config-key エントリーの最大サイズ (単位: バイト)

タイプ

整数

デフォルト

4096

mon_scrub_interval

説明

保存されているチェックサムと、保存されているすべての鍵の計算されたチェックサムを比較して、モニターがストアをスクラブする頻度 (秒単位)

タイプ

整数

デフォルト

3600*24

mon_scrub_max_keys

説明

都度スクラブするキーの最大数

型

整数

デフォルト

100

mon_compact_on_start

説明

ceph-mon の起動時に Ceph Monitor ストアとして使用されるデータベースを圧縮します。手動コンパクションは、通常のコンパクションが機能しない場合に、モニターデータベースを縮小し、パフォーマンスを向上させるのに役立ちます。

型

ブール値

デフォルト

False

mon_compact_on_bootstrap

説明

ブートストラップ時に Ceph Monitor ストアとして使用されるデータベースを圧縮します。ブー

トストラップ後に、モニターはクォーラムを作るためにお互いにプロービングを開始します。クォーラムに参加する前にタイムアウトした場合は、やり直して、再びブートストラップを行います。

型

ブール値

デフォルト

False

mon_compact_on_trim

説明

古い状態をトリミングする際に、あるプレフィックス (paxos を含む) をコンパクト化します。

型

ブール値

デフォルト

True

mon_cpu_threads

説明

モニター上で CPU 負荷の高い作業を行うためのスレッドの数

型

ブール値

デフォルト

True

mon_osd_mapping_pgs_per_chunk

説明

配置グループから OSD へのマッピングをチャンクで計算します。このオプションで、チャンクごとの配置グループ数を指定します。

型

整数

デフォルト

4096

mon_osd_max_split_count

説明

分割を作成させるための「関係する」OSD ごとの最大の PG 数。プールの **pg_num** を増やすと、配置グループは、そのプールを提供するすべての OSD で分割されます。PG を分割する際、極端な倍数は避けるべきです。

型

整数

デフォルト

300

mon_session_timeout

説明

Monitor は、非アクティブなセッションを終了し、この制限時間を超えてアイドル状態を維持します。

型

整数

デフォルト

300

rados_mon_op_timeout

説明

RADOS 操作からのエラーを返す前に、RADOS が Ceph Monitor からの応答を待つ時間 (秒数)。値が 0 の場合は制限がないことを意味します。

型

double

デフォルト

0

第4章 CEPHX 設定リファレンス

cephx プロトコルはデフォルトで有効になっています。暗号認証には多少の計算コストがかかりますが、一般的には非常に低いものです。クライアントとサーバーホストを接続するネットワーク環境が非常に安全で、認証を行う余裕がない場合は、無効にすることができます。ただし、Red Hat は認証の使用を推奨しています。

注記

認証を無効にすると、中間者攻撃によってクライアントとサーバーのメッセージが改ざんされる危険性があり、重大なセキュリティ問題に発展する可能性があります。

4.1. MANUAL (手動)

クラスターを手動でデプロイする場合、モニターを手動でブートストラップし、**client.admin** ユーザーとキーリングを作成する必要があります。Ceph を手動でデプロイするには、ナレッジベースの 記事 を参照してください。モニターのブートストラップの手順は、Chef、Puppet、Juju などのサードパーティーの展開ツールを使用するときに実行する必要がある論理的な手順です。

4.2. CEPHX の有効化と無効化

Cephx を有効にするには、モニターと OSD のキーをデプロイしておく必要があります。Cephx のオン/オフを切り替えるだけの場合は、ブートストラップ手順を繰り返す必要はありません。

4.2.1. Cephx の有効化

cephx が有効な場合には、Ceph はデフォルトの検索パス **/etc/ceph/$cluster.$name.keyring** を含む) でキーリングを探します。Ceph 設定ファイルの **[global]** セクションに **keyring** オプションを追加することで、この場所を上書きすることができますが、これは推奨されません。

認証が無効になっているクラスターで **cephx** を有効にするには、以下の手順を実行します。ご自身またはデプロイメントユーティリティーがすでにキーを生成している場合は、キーの生成に関する手順を省略できます。

1. **client.admin** キーを作成し、クライアントホストのキーのコピーを保存します。

```
ceph auth get-or-create client.admin mon 'allow *' osd 'allow *' -o
/etc/ceph/ceph.client.admin.keyring
```

警告

これにより、既存の **/etc/ceph/client.admin.keyring** ファイルの内容が消去されます。すでにデプロイメントツールがこの作業を行っている場合は、この手順を実行しないでください。

2. モニタークラスター用のキーリングを作成し、モニターシークレットキーを生成します。

```
ceph-authtool --create-keyring /tmp/ceph.mon.keyring --gen-key -n mon. --cap mon 'allow *'
```

3. すべてのモニターの **mon data** ディレクトリーの **ceph.mon.keyring** ファイルにモニターキーリングをコピーします。たとえば、これをクラスター**ceph** の **mon.a** にコピーするには、以下のコマンドを使用します。

```
cp /tmp/ceph.mon.keyring /var/lib/ceph/mon/ceph-a/keyring
```

4. すべての OSD に秘密鍵を生成します。ここで、**{$id}** は OSD 番号です。

```
ceph auth get-or-create osd.{$id} mon 'allow rwx' osd 'allow *' -o /var/lib/ceph/osd/ceph-{$id}/keyring
```

5. デフォルトでは、**cephx** 認証プロトコルは有効になっています。

注記

認証オプションを **none** に設定して **cephx** 認証プロトコルが無効にされていた場合には、Ceph 設定ファイル (**/etc/ceph/ceph.conf**) の **[global]** セクションの下にある以下の行を削除して、**cephx** 認証プロトコルを再度有効にします。

```
auth_cluster_required = none
auth_service_required = none
auth_client_required = none
```

6. Ceph クラスターを起動または再起動します。

重要

cephx を有効にするには、クラスターを完全に再起動する必要があるか、クライアントの I/O が無効になったときにシャットダウンしてから起動する必要があるため、ダウンタイムが必要です。

これらのフラグは、ストレージクラスターを再起動またはシャットダウンする前に設定する必要があります。

```
# ceph osd set noout
# ceph osd set norecover
# ceph osd set norebalance
# ceph osd set nobackfill
# ceph osd set nodown
# ceph osd set pause
```

cephx が有効になり、すべての PG がアクティブかつクリーンな状態になったら、フラグの設定を解除します。

```
# ceph osd unset noout
# ceph osd unset norecover
# ceph osd unset norebalance
# ceph osd unset nobackfill
# ceph osd unset nodown
# ceph osd unset pause
```

4.2.2. Cephx の無効化

以下の手順では、Cephx を無効にする方法を説明します。クラスター環境が比較的安全であれば、認証を実行するための計算コストを相殺することができます。Red Hat では認証を有効にすることを推奨しています。しかし、セットアップやトラブルシューティングの際には、一時的に認証を無効にした方が簡単な場合もあります。

1. Ceph 設定ファイルの **[global]** セクションに以下のオプションを設定して、**cephx** 認証を無効にします。

```
auth_cluster_required = none
auth_service_required = none
auth_client_required = none
```

2. Ceph クラスターを起動または再起動します。

4.3. 設定方法

4.3.1. イネーブルメント

auth_cluster_required

説明

有効な場合、Red Hat Ceph Storage クラスターデーモン (つまり、**ceph-mon** および **ceph-osd**) は相互に認証する必要があります。有効な設定は **cephx** または **none** です。

型

文字列

必須

いいえ

デフォルト

cephx.

auth_service_required

説明

有効にすると、Red Hat Ceph Storage クラスターデーモンは、Ceph サービスにアクセスするために、Ceph クライアントが Red Hat Ceph Storage クラスターと認証することを要求します。有効な設定は **cephx** または **none** です。

型

文字列

必須

いいえ

デフォルト

cephx.

auth_client_required

説明

有効にすると、Ceph クライアントは、Red Hat Ceph Storage クラスターが Ceph クライアントと認証することを要求します。有効な設定は **cephx** または **none** です。

型

文字列

必須

いいえ

デフォルト

cephx.

4.3.2. 鍵

認証が有効な Ceph を実行する場合には、Ceph Storage クラスターにアクセスするために **ceph** 管理コマンドおよび Ceph クライアントに認証キーが必要です。

ceph 管理コマンドおよびクライアントにこれらの鍵を提供する最も一般的な方法は、**/etc/ceph/** ディレクトリーの下に Ceph キーリングを追加することです。ファイル名は通常 **ceph.client.admin.keyring** または **$cluster.client.admin.keyring** です。**/etc/ceph/** ディレクトリーにキーリングを含める場合は、Ceph 設定ファイルで **keyring** エントリーを指定する必要はありません。

Red Hat は、**client.admin** キーが含まれるため、Red Hat Ceph Storage クラスターのキーリングファイルを管理コマンドを実行するノードにコピーすることを推奨します。それを行うには、**root** で以下のコマンドを実行します。

```
# scp <user>@<hostname>:/etc/ceph/ceph.client.admin.keyring /etc/ceph/ceph.client.admin.keyring
```

<user> をホストで使用されているユーザー名に置き換え、**client.admin** キーを使用し、**<hostname>** をそのホストのホスト名に置き換えます。

注記

ceph.keyring ファイルに、クライアントマシンに適切なパーミッションが設定されていることを確認します。

推奨されていない **key** 設定を使用して、Ceph 設定ファイルにキー自体を指定したり、**keyfile** 設定を使用してキーファイルへのパスを指定することができます。

keyring

詳細

キーリングファイルのパス

型

文字列

必須

いいえ

デフォルト

/etc/ceph/$cluster.$name.keyring,/etc/ceph/$cluster.keyring,/etc/ceph/keyring,/etc/ceph/keyring.bin

keyfile

詳細

キーファイル (つまり、キーのみを含むファイル) へのパス

型

文字列

必須

いいえ

デフォルト

なし

key

詳細

キー (つまり、キーそのもののテキスト文字列)。推奨されません。

型

文字列

必須

いいえ

デフォルト

None

4.3.3. デーモンキーリング

管理ユーザーやデプロイメントツールは、ユーザーキーリングの生成と同じ方法で、デーモンキーリングを生成することがあります。デフォルトでは、Ceph はデーモンのキーリングをデータディレクトリー内に保存します。デフォルトのキーリングの場所と、デーモンが機能するために必要な機能を以下に示します。

ceph-mon

場所

$mon_data/keyring

ケイパビリティー

mon 'allow *'

ceph-osd

場所

$osd_data/keyring

ケイパビリティー

mon 'allow profile osd' osd 'allow *'

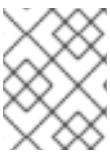
radosgw

場所

$rgw_data/keyring

ケイパビリティー

mon 'allow rwx' osd 'allow rwx'



注記

モニターのキーリング (つまり **mon.**) にはキーが含まれていますが、機能は含まれておらず、クラスター **auth** データベースの一部ではありません。

デーモンデータのディレクトリの位置は、デフォルトでは以下の形式のディレクトリになります。

```
/var/lib/ceph/$type/$cluster-$id
```

たとえば、**osd.12** は以下のようになります。

```
/var/lib/ceph/osd/ceph-12
```

これらの場所を上書きすることもできますが、お勧めできません。

4.3.4. 署名

Red Hat では、最初の認証のために設定されたセッションキーを使用して、Ceph がエンティティ間のすべての進行中のメッセージを認証することを推奨しています。

Ceph 認証の他の部分と同様に、Ceph はきめ細かい制御を提供するため、クライアントと Ceph 間のサービスメッセージの署名を有効または無効にしたり、Ceph デーモン間のメッセージの署名を有効または無効にしたりできます。

cephx_require_signatures

説明

true に設定すると、Ceph は、Ceph クライアントと Red Hat Ceph Storage クラスターの間、および Red Hat Ceph Storage クラスターを設定するデーモン間のすべてのメッセージトラフィックで署名を必要とします。

タイプ

ブール値

必須

いいえ

デフォルト

false

cephx_cluster_require_signatures

説明

true に設定した場合には、Ceph では、Red Hat Ceph Storage クラスターを構成する Ceph デーモン間のすべてのメッセージトラフィックに対する署名が必要です。

タイプ

ブール値

必須

いいえ

デフォルト

false

cephx_service_require_signatures

説明

true に設定した場合には、Ceph クライアントと Red Hat Ceph Storage クラスター間のすべてのメッセージトラフィックに対する署名が必要です。

タイプ

ブール値

必須

いいえ

デフォルト

false

cephx_sign_messages

詳細

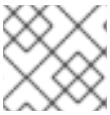
Ceph のバージョンがメッセージ署名をサポートしている場合、Ceph はすべてのメッセージに署名し、メッセージが偽装されないようにします。

型

ブール値

デフォルト

true



注記

Ceph のカーネルモジュールは、まだ署名をサポートしていません。

4.3.5. Time to Live

auth_service_ticket_ttl

説明

Red Hat Ceph Storage クラスターが Ceph クライアントに認証用のチケットを送信すると、クラスターはそのチケットに生存時間を割り当てます。

タイプ

double

デフォルト

60*60

第5章 プール、PG、および CRUSH 設定リファレンス

プールを作成し、プールの配置グループの数を設定するとき、特にデフォルトをオーバーライドしない場合、Ceph はデフォルト値を使用します。Red Hat では、いくつかのデフォルトを上書きすることをお勧めします。具体的には、プールのレプリカサイズを設定し、デフォルトの配置グループ数を上書きします。これらの値は、pool コマンドの実行時に設定できます。Ceph 設定ファイルの **[global]** セクションに新規のものを追加して、デフォルト値を上書きすることもできます。

```
[global]

# By default, Ceph makes 3 replicas of objects. If you want to set 4
# copies of an object as the default value--a primary copy and three replica
# copies--reset the default values as shown in 'osd pool default size'.
# If you want to allow Ceph to write a lesser number of copies in a degraded
# state, set 'osd pool default min size' to a number less than the
# 'osd pool default size' value.

osd_pool_default_size = 4  # Write an object 4 times.
osd_pool_default_min_size = 1 # Allow writing one copy in a degraded state.

# Ensure you have a realistic number of placement groups. We recommend
# approximately 100 per OSD. E.g., total number of OSDs multiplied by 100
# divided by the number of replicas (i.e., osd pool default size). So for
# 10 OSDs and osd pool default size = 4, we'd recommend approximately
# (100 * 10) / 4 = 250.

osd_pool_default_pg_num = 250
osd_pool_default_pgp_num = 250
```

5.1. 設定

mon_allow_pool_delete

詳細

モニターがプールを削除することができます。RHCS 3 以降のリリースでは、データ保護のための追加措置として、モニターはデフォルトでプールを削除できません。

型

ブール値

デフォルト

false

mon_max_pool_pg_num

詳細

プールあたりの配置グループの最大数

型

整数

デフォルト

65536

mon_pg_create_interval

詳細

同じ Ceph OSD デーモンでの PG 作成の間の秒数

型

浮動小数点 (Float)

デフォルト

30.0

mon_pg_stuck_threshold

詳細

PG がスタックしていると判断できるまでの秒数

型

32 ビット整数

デフォルト

300

mon_pg_min_inactive

詳細

Ceph は、**mon_pg_stuck_threshold** より長く非アクティブのままの PG の数がこの設定を超える場合に、クラスターログに **HEALTH_ERR** ステータスを発行します。デフォルト設定は 1 つの PG です。正数以外の数値を指定すると、この設定は無効になります。

型

整数

デフォルト

1

mon_pg_warn_min_per_osd

詳細

Ceph は、クラスター内の OSD ごとの PG の平均数がこの設定よりも小さい場合に、クラスターログで **HEALTH_WARN** ステータスを発行します。正数以外の数値を指定すると、この設定は無効になります。

型

整数

デフォルト

30

mon_pg_warn_max_per_osd

詳細

Ceph は、クラスター内の OSD ごとの PG の平均数がこの設定よりも大きい場合に、クラスターログの **HEALTH_WARN** ステータスを発行します。正数以外の数値を指定すると、この設定は無効になります。

型

整数

デフォルト

300

mon_pg_warn_min_objects

詳細

クラスター内のオブジェクトの総数がこの数以下の場合は警告を発生しません。

型

整数

デフォルト

1000

mon_pg_warn_min_pool_objects

詳細

オブジェクト数がこの数以下のプールには警告を発生しません。

型

整数

デフォルト

1000

mon_pg_check_down_all_threshold

詳細

down OSD のしきい値 (パーセント) で、Ceph はすべての PG をチェックして、それらがスタックまたは古くなっていることを確認します。

型

浮動小数点 (Float)

デフォルト

0.5

mon_pg_warn_max_object_skew

詳細

プール内のオブジェクトの平均数 **mon pg warn max object skew** を超える場合、Ceph はクラスターログで **HEALTH_WARN** ステータスを発行します。正数以外の数値を指定すると、この設定は無効になります。

型

浮動小数点 (Float)

デフォルト

10

mon_delta_reset_interval

詳細

Ceph が PG デルタをゼロにリセットするまでの非アクティブ時の秒数。Ceph は、各プールの使用済み容量のデルタを追跡し、管理者がリカバリーの進捗状況やパフォーマンスを評価するのに役立てます。

型

整数

デフォルト

10

mon_osd_max_op_age

詳細

HEALTH_WARN ステータスを発行する前に操作が完了するまでの最大期間 (秒単位)。

型

浮動小数点 (Float)

デフォルト

32.0

osd_pg_bits

詳細

Ceph OSD デーモンごとの配置グループのビット数

型

32 ビット整数

デフォルト

6

osd_pgp_bits

詳細

配置目的の配置グループ (PGP) の Ceph OSD デーモンあたりのビット数

型

32 ビット整数

デフォルト

6

osd_crush_chooseleaf_type

詳細

CRUSH ルールで **chooseleaf** に使用するバケットタイプ。名前ではなく従来のランクを使用します。

型

32 ビット整数

デフォルト

1.通常は、1 つまたは複数の Ceph OSD デーモンを含むホストです。

osd_pool_default_crush_replicated_ruleset

詳細

レプリケートされたプールを作成する際に使用するデフォルトの CRUSH ルールセット

型

8 ビット整数

デフォルト

0

osd_pool_erasure_code_stripe_unit

詳細

イレイジャーコード化されたプールのオブジェクトストライプのチャンクのデフォルトサイズをバイト単位で設定します。サイズ S のすべてのオブジェクトは N ストライプとして格納され、各データチャンクは **stripe unit** バイトを受け取ります。**N * stripe unit** バイトの各ストライプは、個別にエンコード/エンコードされます。このオプションは、イレイジャーコードプロファイルの **stripe_unit** 設定で上書きできます。

型

32 ビット符号なし整数

デフォルト

4096

osd_pool_default_size

詳細

プール内のオブジェクトのレプリカ数を設定します。デフォルト値は、**ceph osd pool set {pool-name} size {size}** と同じです。

型

32 ビット整数

デフォルト

3

osd_pool_default_min_size

詳細

プール内のオブジェクトに対して、クライアントへの書き込み操作を確認するための、書き込み済みレプリカの最小数を設定します。最小値が満たされていない場合、Ceph はクライアントへの書き込みを確認しません。この設定により、**degraded** モードで動作している場合にレプリカの最小数を確保できます。

型

32 ビット整数

デフォルト

0 (これは、特定の最小値がないことを意味します)**0** の場合、最小は **size - (size** / **2)** になります。

osd_pool_default_pg_num

詳細

プールの配置グループのデフォルト数。デフォルト値は、**mkpool** で **pg_num** と同じです。

型

32 ビット整数

デフォルト

8

osd_pool_default_pgp_num

詳細

プールに対する配置の配置グループのデフォルト数です。デフォルト値は、**mkpool** で **pgp_num** と同じです。PG と PGP は等しいはずです (今のところ)。

タイプ

32 ビット整数

デフォルト

8

osd_pool_default_flags

詳細

新しいプールのデフォルトフラグ

型

32 ビット整数

デフォルト

0

osd_max_pgls

詳細

リストアップする配置グループの最大数。大きな数を要求するクライアントは、Ceph OSD デーモンを拘束できます。

型

未署名の 64 ビット整数

デフォルト

1024

備考

デフォルトで問題ありません。

osd_min_pg_log_entries

詳細

ログファイルをトリミングする際に維持する配置グループログの最小数

型

32 ビット符号なし整数

デフォルト

1000

osd_default_data_pool_replay_window

説明

クライアントが要求を再生するのを OSD が待機する時間 (秒単位)。

タイプ

32 ビット整数

デフォルト

45

第6章 OSD 設定リファレンス

Ceph 設定ファイルで Ceph OSD を設定できますが、Ceph OSD はデフォルト値と非常に最小限の設定を使用できます。最小限の Ceph OSD 設定では、**osd journal size** および **osd host** オプションを設定し、その他のほとんどすべてにデフォルト値を使用します。

Ceph OSD は、次の規則を使用して **0** から始まる増分方式で数値的に識別されます。

```
osd.0
osd.1
osd.2
```

設定ファイルでは、設定ファイルの **osd** セクションに設定設定を追加することで、クラスター内のすべての Ceph OSD の設定を指定できます。特定の Ceph OSD (たとえば、osd **host**) に設定を直接追加するには、Ceph 設定ファイルのその OSD のみに固有のセクションに入力します。以下に例を示します。

```
[osd]
osd journal size = 1024

[osd.0]
osd host = osd-host-a

[osd.1]
osd host = osd-host-b
```

6.1. 一般設定

以下の設定は、Ceph OSD の ID を提供し、データとジャーナルへのパスを決定します。通常、Ceph デプロイメントスクリプトは UUID を自動的に生成します。

重要

Red Hat は、データまたはジャーナルのデフォルトパスを変更することをお勧めしません。後で Ceph をトラブルシューティングすることがより困難になるためです。

ジャーナルサイズは、予想されるドライブ速度に **filestore max sync interval** オプションの値を掛けた積の少なくとも 2 倍である必要があります。ただし、最も一般的な方法は、ジャーナルドライブ (多くの場合 SSD) をパーティション分割し、Ceph がパーティション全体をジャーナルに使用するようにマウントすることです。

osd_uuid

詳細

Ceph OSD の Universally Unique Identifier (UUID)

型

UUID

デフォルト

UUID

備考

osd uuid は単一の Ceph OSD に適用されます。 **fsid** はクラスター全体に適用されます。

osd_data

詳細

OSD のデータへのパスCeph のデプロイ時にディレクトリーを作成する必要があります。OSD データ用のドライブをこのマウントポイントにマウントします。Red Hat は、デフォルトを変更することをお勧めしません。

タイプ

文字列

デフォルト

/var/lib/ceph/osd/$cluster-$id

osd_max_write_size

詳細

書き込みの最大サイズ (メガバイト)

型

32 ビット整数

デフォルト

90

osd_client_message_size_cap

詳細

メモリー上で許可される最大のクライアントデータメッセージ

型

64 ビット整数未署名

デフォルト

500 MB のデフォルト**500*1024L*1024L**

osd_class_dir

詳細

RADOS クラスのプラグインのクラスパス

型

文字列

デフォルト

$libdir/rados-classes

6.2. ジャーナル設定

デフォルトでは、Ceph は、Ceph OSD のジャーナルを次のパスに保存することを想定しています。

```
/var/lib/ceph/osd/$cluster-$id/journal
```

パフォーマンスの最適化を行わない場合、Ceph はジャーナルを Ceph OSD のデータと同じディスクに保存します。パフォーマンスが最適化された Ceph OSD は、別のディスクを使用してジャーナルデータを保存できます。たとえば、ソリッドステートドライブは高性能のジャーナリングを提供します。

ジャーナルサイズは、**filestore max sync interval** と予想されるスループットのプロダクトを検索して、プロダクトを 2 (2) で乗算する必要があります。

```
osd journal size = <2 * (expected throughput * filestore max sync interval)>
```

予想されるスループットの数値には、予想されるディスクスループット (つまり、持続的なデータ転送速度) とネットワークスループットが含まれている必要があります。たとえば、7200 RPM のディスクは約 100 MB/秒になる可能性があります。ディスクとネットワークのスループットの **min()** を取得すると、妥当な予想スループットが得られるはずです。一部のユーザーは、10GB のジャーナルサイズから始めます。以下に例を示します。

```
osd journal size = 10000
```

警告

OSD のジャーナルを正しくサイジングすることが重要になります。小さなジャーナルを使用すると、OSD 障害が発生した場合の回復が遅くなります。ジャーナルの圧力を許容レベルに維持して安定した回復を行うには、回復スレッドの数を減らす必要があります。また、ファイルストアへのトランザクションのコミットは遅くなり、キューに入れられたトランザクションサイズがジャーナルサイズよりも大きい場合、ファイルストアがハングする可能性があります。

osd_journal

説明

OSD のジャーナルへのパス。これは、ファイルまたはブロックデバイス (SSD のパーティションなど) へのパスの場合があります。ファイルの場合は、それを格納するディレクトリーを作成する必要があります。**OSD data** ドライブとは別のドライブを使用することをお勧めします。

タイプ

文字列

デフォルト

/var/lib/ceph/osd/$cluster-$id/journal

osd_journal_size

説明

メガバイト単位のジャーナルのサイズ。これが 0 で、ジャーナルがブロックデバイスの場合、ブロックデバイス全体が使用されます。ジャーナルがブロックデバイスの場合、これは無視され、ブロックデバイス全体が使用されます。

タイプ

32 ビット整数

デフォルト

5120

推奨

1GB で始まります。予想される速度にファイルストアの **filestore max sync interval** を掛けた積の少なくとも 2 倍である必要があります。

6.3. スクラブ

Ceph は、オブジェクトの複数のコピーを作成するだけでなく、配置グループをスクラビングすることでデータの整合性を確保します。Ceph のスクラブは、オブジェクトストレージ層の **fsck** コマンドに似ています。

各配置グループについて、Ceph はすべてのオブジェクトのカタログを生成し、各プライマリーオブジェクトとそのレプリカを比較して、オブジェクトの欠落や不一致がないことを確認します。

ライトスクラビング (毎日) では、オブジェクトのサイズや属性をチェックします。ディープスクラビング (毎週) は、データを読み込んでチェックサムでデータの整合性を確保します。

スクラビングはデータの整合性を保つために重要ですが、パフォーマンスを低下させる可能性があります。以下の設定を調整して、スクラブ動作を増減させます。

osd_max_scrubs

詳細

Ceph OSD ごとの同時スクラブ操作の最大数

型

32 ビット整数

デフォルト

1

osd_scrub_thread_timeout

詳細

スクラブスレッドがタイムアウトするまでの最大時間 (秒単位)

型

32 ビット整数

デフォルト

60

osd_scrub_finalize_thread_timeout

詳細

スクラブ最終スレッドがタイムアウトするまでの最大時間 (秒単位)

型

32 ビット整数

デフォルト

60*10

osd_scrub_begin_hour

詳細

軽いスクラブや深いスクラブを始めることができる最も早い時間。これは、スクラビングの時間枠を定義するために **osd scrub end hour** パラメーターと共に使用し、スクラビングをオフピーク時間に制限できるようにします。設定は、24 時間サイクルの時間を指定するために整数を取ります。たとえば、**0** は午前 12:01 から午前 1:00 までを表し、13 は午後 1:01 から午後 2:00 までを表します。

型

32 ビット整数

デフォルト

0 (午前 12:01 から 1:00)

osd_scrub_end_hour

詳細

軽いスクラブや深いスクラブを始めることができる最も遅い時間。これは、**osd scrub begin hour** パラメーターとともに使用してスクラブタイムウィンドウを定義し、スクラブをオフピーク時間に制限します。設定は、24 時間サイクルの時間を指定するために整数を取ります。たとえば、**0** は午前 12:01 から午前 1:00 までを表し、13 は午後 1:01 から午後 2:00 までを表します。**end** 時間は、**begin** 時間よりも大きくなければなりません。

型

32 ビット整数

デフォルト

24 (午後 11:01 から午前 12:00)

osd_scrub_load_threshold

詳細

最大の負荷。(**getloadavg()** 関数で定義された) システムの負荷がこの数値よりも大きい場合、Ceph はスクラブを実行しません。デフォルトは **0.5** です。

型

浮動小数点 (Float)

デフォルト

0.5

osd_scrub_min_interval

説明

Red Hat Ceph Storageクラスターの負荷が低いときに、Ceph OSD をスクラブする最小の間隔 (秒単位)

タイプ

浮動小数点 (Float)

デフォルト

1 日 1 回。**60*60*24**

osd_scrub_max_interval

詳細

クラスター負荷に関わらず Ceph OSD をスクラビングする最大の間隔 (秒単位)。

型

浮動小数点 (Float)

デフォルト

1 週間に 1 回になります。**7*60*60*24**

osd_scrub_interval_randomize_ratio

詳細

比率を取り、**osd scrub min interval** および **osd scrub max interval** の間隔の間でスケジュールされたスクラブをランダム化します。

型

浮動小数点 (Float)

デフォルト

0.5。

mon_warn_not_scrubbed

詳細

スクラブされていない PG について警告する **osd_scrub_interval** からの秒数。

型

整数

デフォルト

0 (警告なし)。

osd_scrub_chunk_min

詳細

オブジェクトストアは、ハッシュの境界で終わるチャンクに分割されています。チャンキースクラブの場合、Ceph はオブジェクトを 1 チャンクずつスクラブし、そのチャンクへの書き込みをブロックします。**osd scrub chunk min** 設定は、スクラビングするチャンクの最小数を表します。

型

32 ビット整数

デフォルト

5

osd_scrub_chunk_max

詳細

スクラブするチャンクの最大数

型

32 ビット整数

デフォルト

25

osd_scrub_sleep

詳細

ディープスクラブ操作の間のスリープ時間

型

浮動小数点 (Float)

デフォルト

0 (またはオフ)

osd_scrub_during_recovery

詳細

リカバリー時のスクラブを可能にします。

型

ブール (Bool)

デフォルト

false

osd_scrub_invalid_stats

詳細

無効と判定された統計情報を修正するために、強制的に追加のスクラブを実行します。

型

ブール (Bool)

デフォルト

true

osd_scrub_priority

詳細

クライアント I/O に対するスクラブ操作のキューの優先順位を制御します。

型

32 ビット符号なし整数

デフォルト

5

osd_scrub_cost

詳細

キューのスケジューリングのために、スクラブ操作のコストをメガバイト単位で表したもの。

型

32 ビット符号なし整数

デフォルト

50 << 20

osd_deep_scrub_interval

詳細

すべてのデータを完全に読み込むディープスクラビングのための間隔。**osd scrub load threshold** パラメーターは、この設定には影響を与えません。

型

浮動小数点 (Float)

デフォルト

1 週間に 1 回になります。**60*60*24*7**

osd_deep_scrub_stride

詳細

デープスクラブを実施する際の読み取りサイズ

型

32 ビット整数

デフォルト

512 KB。**524288**

mon_warn_not_deep_scrubbed

詳細

スクラビングされていない PG について警告する **osd_deep_scrub_interval** からの秒数。

型

整数

デフォルト

0 (警告なし)。

osd_deep_scrub_randomize_ratio

詳細

スクラブが無作為にディープスクラビングになる変化 (**osd_deep_scrub_interval** が経過する可能性も)

型

浮動小数点 (Float)

デフォルト

0.15 または 15%。

osd_deep_scrub_update_digest_min_age

詳細

スクラブがオブジェクト全体のダイジェストを更新するまでに、オブジェクトが何秒経過していなければならないか。

型

整数

デフォルト

120 (2 時間)。

6.4. 操作

操作設定では、リクエストを処理するためのスレッド数を設定できます。

デフォルトでは、Ceph はタイムアウトが 30 秒の 2 つのスレッドを使用し、操作がこれらの時間パラメーター内に完了しない場合は 30 秒の苦情時間を設定します。クライアント操作と回復操作の間に操作の優先順位の重みを設定して、回復中の最適なパフォーマンスを確保します。

osd_op_num_shards

詳細

クライアント操作のためのシャード数

型

32 ビット整数

デフォルト

0

osd_op_num_threads_per_shard

詳細

クライアント操作のためのシャードあたりのスレッド数

型

32 ビット整数

デフォルト

0

osd_op_num_shards_hdd

詳細

HDD 操作のためのシャード数

型

32 ビット整数

デフォルト

5

osd_op_num_threads_per_shard_hdd

詳細

HDD 操作のためのシャードあたりのスレッド数

型

32 ビット整数

デフォルト

1

osd_op_num_shards_ssd

詳細

SSD 操作のためのシャード数

型

32 ビット整数

デフォルト

8

osd_op_num_threads_per_shard_ssd

詳細

SSD 操作のためのシャードあたりのスレッド数

型

32 ビット整数

デフォルト

2

osd_client_op_priority

詳細

クライアントの操作に設定されている優先順位。これは、**osd recovery op priority** と相対的になります。

型

32 ビット整数

デフォルト

63

有効な範囲

1-63

osd_recovery_op_priority

詳細

復元の操作に設定されている優先順位。これは、**osd client op priority** と相対的になります。

型

32 ビット整数

デフォルト

3

有効な範囲

1-63

osd_op_thread_timeout

詳細

Ceph OSD 操作スレッドのタイムアウト (秒単位)

型

32 ビット整数

デフォルト

30

osd_op_complaint_time

詳細

指定された秒数が経過すると、クレームに値する操作になります。

型

浮動小数点 (Float)

デフォルト

30

osd_disk_threads

詳細

スクラビングやスナップトリミングなど、バックグラウンドでのディスクを多用する OSD 操作に使用されるディスクスレッドの数

型

32 ビット整数

デフォルト

1

osd_disk_thread_ioprio_class

詳細

ディスクスレッドに **ioprio_set(2)** I/O スケジューリング **class** を設定します。設定可能な値は以下のとおりです。

- **idle**
- **be**
- **rt**

idle クラスは、ディスクスレッドの優先度が OSD 内の他のどのスレッドよりも低いことを意味します。これは、クライアント操作の処理で忙しい OSD のスクラブを遅くするのに役立ちます。

be クラスはデフォルトであり、OSD 内の他のすべてのスレッドと同じ優先度です。

rt クラスは、ディスクスレッドが OSD の他のすべてのスレッドよりも優先されます。この機能は、スクラブが必要で、クライアントの操作を犠牲にしてでもスクラブを行う必要がある場合に有効です。

型

文字列

デフォルト

空の文字列

osd_disk_thread_ioprio_priority

詳細

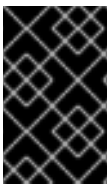
ディスクスレッドの **ioprio_set(2)** I/O スケジューリングの **priority** を 0 (最高) から 7 (最低) に設定します。指定のホストの OSD がすべて **idle** クラスで、コントローラーが輻輳により I/O に対して競合している場合には、1 つの OSD のディスクスレッド優先度を 7 に下げ、優先度が 0 の別の OSD をよりスクラビングすることが可能になります。+

タイプ

0 - 7 の範囲の整数で、使用しない場合は -1

デフォルト

-1



重要

osd disk thread ioprio class および **osd disk thread ioprio priority** オプションは、両方がデフォルト値以外の値に設定されている場合にのみ使用されます。また、Linux Kernel CFQ スケジューラでのみ動作します。

osd_op_history_size

詳細

追跡する完了した操作の最大数

型

32 ビット未署名の整数

デフォルト

20

osd_op_history_duration

詳細

追跡する最も古い完了した操作

型

32 ビット未署名の整数

デフォルト

600

osd_op_log_threshold

詳細

一度に表示する操作ログの数

型

32 ビット整数

デフォルト

5

osd_op_timeout

詳細

実行中の OSD 操作がタイムアウトするまでの時間 (秒)

型

整数

デフォルト

0

重要

クライアントが結果に対応できない限り、**osd op timeout** オプションを設定しないでください。例えば、仮想マシン上で動作するクライアントにこのパラメータを設定すると、仮想マシンがこのタイムアウトをハードウェアの故障と解釈するため、データの破損につながる可能性があります。

6.5. バックフィル

Ceph OSD をクラスターに追加したり、クラスターから削除したりすると、CRUSH アルゴリズムは、配置グループを Ceph OSD に移動させたり、Ceph OSD から移動させたりしてバランスを回復させ、クラスターのバランスを取り戻します。配置グループとそれに含まれるオブジェクトを移行するプロセスは、クラスターの運用パフォーマンスを大幅に低下させます。運用パフォーマンスを維持するために、Ceph はこの移行を「バックフィル」プロセスで実行します。これにより、Ceph はバックフィル操作をデータの読み取りまたは書き込みの要求よりも低い優先度に設定できます。

osd_max_backfills

詳細

1 つの OSD に対して、または 1 つの OSD から許容されるバックフィル操作の最大数

型

64 ビット未署名の整数

デフォルト

1

osd_backfill_scan_min

詳細

バックフィルスキャン 1 回あたりのオブジェクトの最小数

型

32 ビット整数

デフォルト

64

osd_backfill_scan_max

詳細

バックフィルスキャン 1 回あたりのオブジェクトの最大数

型

32 ビット整数

デフォルト

512

osd_backfillfull_ratio

説明

Ceph OSD のフル比率がこの値以上の場合、バックフィル要求の受け入れを拒否します。

型

浮動小数点 (Float)

デフォルト

0.85

osd_backfill_retry_interval

詳細

バックフィル要求を再試行するまでの待ち時間 (秒数)

型

double

デフォルト

10.0

6.6. OSD マップ

OSD マップは、クラスターで稼働している OSD デーモンを反映します。時間が経つにつれ、マップエポックの数が増えます。Ceph は以下の設定を提供し、Ceph が実行すると共に OSD マップが大きくなるようにします。

osd_map_dedup

詳細

OSD マップの重複の削除を有効にします。

型

ブール値

デフォルト

true

osd_map_cache_size

詳細

OSD マップキャッシュのサイズ (メガバイト)

型

32 ビット整数

デフォルト

50

osd_map_cache_bl_size

詳細

OSD デーモンのメモリー内 OSD マップキャッシュのサイズ

型

32 ビット整数

デフォルト

50

osd_map_cache_bl_inc_size

詳細

OSD デーモンのメモリー内 OSD マップキャッシュの増分サイズ

型

32 ビット整数

デフォルト

100

osd_map_message_max

詳細

MOSDMap メッセージごとに許容される最大のマップエントリー数

型

32 ビット整数

デフォルト

40

6.7. 復元

クラスターが起動したとき、または Ceph OSD が予期せず終了して再起動したとき、OSD は書き込み操作を行う前に他の Ceph OSD とのピアリングを開始します。

Ceph OSD がクラッシュしてオンラインに戻ると、通常、配置グループのオブジェクトのより新しいバージョンが含まれる他の Ceph OSD との同期が取れなくなります。このような場合、Ceph OSD はリカバリーモードに入り、データの最新コピーを取得してマップを最新の状態に戻そうとします。

Ceph OSD が停止していた時間によっては、OSD のオブジェクトや配置グループが大幅に古くなっている可能性があります。また、障害ドメイン (例: ラックなど) ダウンした場合、複数の Ceph OSD が同時にオンラインに戻る可能性があります。そのため、復旧作業には時間とリソースが必要になります。

運用パフォーマンスを維持するために、Ceph はリカバリー要求数、スレッド数、およびオブジェクトチャンクサイズを制限してリカバリーを実行し、これにより Ceph は劣化した状態でも適切なパフォーマンスを発揮することができます。

osd_recovery_delay_start

詳細

ピアリングが完了すると、Ceph はオブジェクトの回復を開始する前に、指定された秒数だけ遅延します。

型

浮動小数点 (Float)

デフォルト

0

osd_recovery_max_active

詳細

OSD ごとに一度のアクティブな復旧要求の数。リクエストが増えれば復旧も早くなりますが、その分クラスターへの負荷も大きくなります。

型

32 ビット整数

デフォルト

3

osd_recovery_max_chunk

詳細

復元したデータチャンクをプッシュする際の最大サイズ

型

64 ビット整数未署名

デフォルト

8 << 20

osd_recovery_threads

詳細

データを復元するためのスレッド数

型

32 ビット整数

デフォルト

1

osd_recovery_thread_timeout

詳細

復元スレッドがタイムアウトするまでの最大時間 (秒単位)

型

32 ビット整数

デフォルト

30

osd_recover_clone_overlap

詳細

復元時のクローンのオーバーラップを保持します。常に **true** に設定する必要があります。

型

ブール値

デフォルト

true

6.8. その他

osd_snap_trim_thread_timeout

詳細

スナップトリムスレッドがタイムアウトするまでの最大時間 (秒単位)

型

32 ビット整数

デフォルト

60*60*1

osd_pg_max_concurrent_snap_trims

詳細

PG ごとの並列スナップトリムの最大数。PG ごとに何個のオブジェクトを一度にトリミングするかを制御します。

型

32 ビット整数

デフォルト

2

osd_snap_trim_sleep

詳細

PG が発行する各トリム操作の間にスリープを挿入します。

型

32 ビット整数

デフォルト

0

osd_max_trimming_pgs

詳細

トリミング PG の最大数

型

32 ビット整数

デフォルト

2

osd_backlog_thread_timeout

詳細

バックログスレッドがタイムアウトするまでの最大時間 (秒単位)

型

32 ビット整数

デフォルト

60*60*1

osd_default_notify_timeout

詳細

OSD デフォルト通知のタイムアウト (単位: 秒)

型

32 ビット符号なし整数

デフォルト

30

osd_check_for_log_corruption

詳細

ログファイルが破損していないか確認します。計算量が多くなる可能性があります。

型

ブール値

デフォルト

false

osd_remove_thread_timeout

詳細

OSD 削除スレッドがタイムアウトするまでの最大時間 (秒単位)

型

32 ビット整数

デフォルト

60*60

osd_command_thread_timeout

詳細

コマンドスレッドがタイムアウトするまでの最大時間 (秒単位)

型

32 ビット整数

デフォルト

10*60

osd_command_max_records

詳細

失ったオブジェクトを返す際の数を制限します。

型

32 ビット整数

デフォルト

256

osd_auto_upgrade_tmap

詳細

古いオブジェクトの **omap** に **tmap** を使用します。

型

ブール値

デフォルト

true

osd_tmapput_sets_users_tmap

詳細

デバッグにだけ **tmap** を使用します。

型

ブール値

デフォルト

false

osd_preserve_trimmed_log

詳細

トリミングされたログファイルは保持されますが、より多くのディスク容量を使用します。

型

ブール値

デフォルト

false

rados_osd_op_timeout

詳細

RADOS 操作からのエラーを返す前に、RADOS が OSD からの応答を待つ時間 (秒数)。値が 0 の場合は制限がないことを意味します。

型

double

デフォルト

0

第7章 モニターと OSD の相互作用の設定

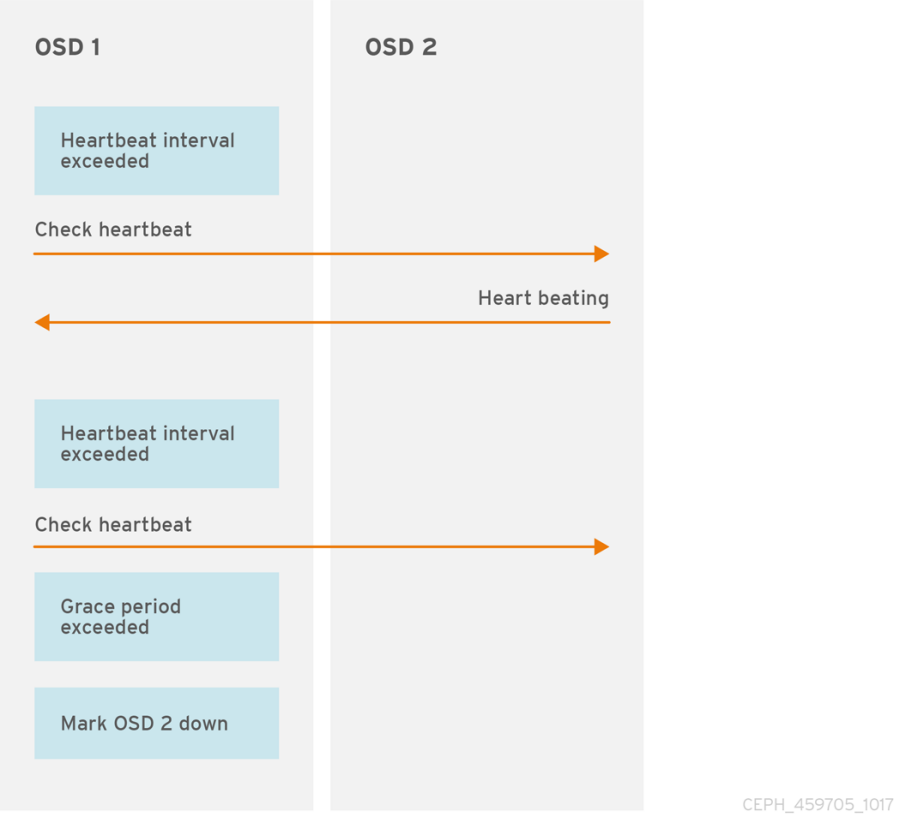
Ceph の初期設定が完了したら、Ceph をデプロイして実行することができます。**ceph health**、**ceph -s** などのコマンドを実行すると、Ceph Monitor は Ceph Storage クラスターの現在の状態を報告します。Ceph Monitor は、各 Ceph OSD デーモンからのレポートを要求し、隣接する Ceph OSD デーモンのステータスに関するレポートを Ceph OSD デーモンから受信することによって、Ceph Storage Cluster について認識します。Ceph Monitor がレポートを受信しない場合、または Ceph Storage Cluster の変更に関するレポートを受信する場合、Ceph Monitor は Ceph Cluster Map のステータスを更新します。

Ceph は、Ceph Monitor と Ceph OSD Daemon の相互作用のための合理的なデフォルト設定を提供します。ただし、デフォルト値を上書きできます。以下のセクションでは、Ceph Storage Cluster を監視する目的で Ceph Monitor と Ceph OSD Daemon がどのように相互作用するかについて説明します。

7.1. OSD はハートビートをチェック

各 Ceph OSD デーモンは、6 秒ごとに他の Ceph OSD デーモンのハートビートをチェックします。ハートビートの間隔を変更するには、Ceph 設定ファイルの **[osd]** セクションに **osd heartbeat interval** 設定を追加するか、ランタイム時にその値を変更します。

隣接する Ceph OSD デーモンが 20 秒の猶予期間内にハートビートパケットを送信しない場合、Ceph OSD デーモンは隣接する Ceph OSD デーモンが **down** していると見なし、それを Ceph Monitor に報告して、Ceph クラスターマップを更新します。この猶予期間を変更するには、Ceph 設定ファイルの **[osd]** セクションに **osd heartbeat grace** 設定を追加するか、ランタイム時にの値を設定します。



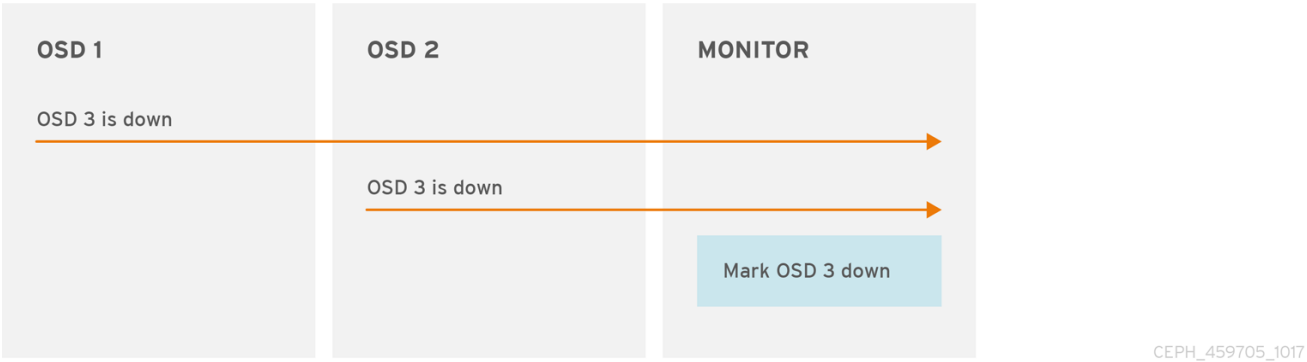
7.2. OSD レポートダウン OSD

デフォルトでは、**異なるホスト** の 2 つの Ceph OSD デーモンは、報告された Ceph OSD デーモンが **down** していることを Ceph モニターが確認する前に、別の CephOSD デーモンが **down** していることを Ceph モニターに報告する必要があります。

しかし、障害を報告するすべての OSD が、ラック内の異なるホストに設置されており、スイッチ不良により OSD 間の接続に問題が生じる場合があります。

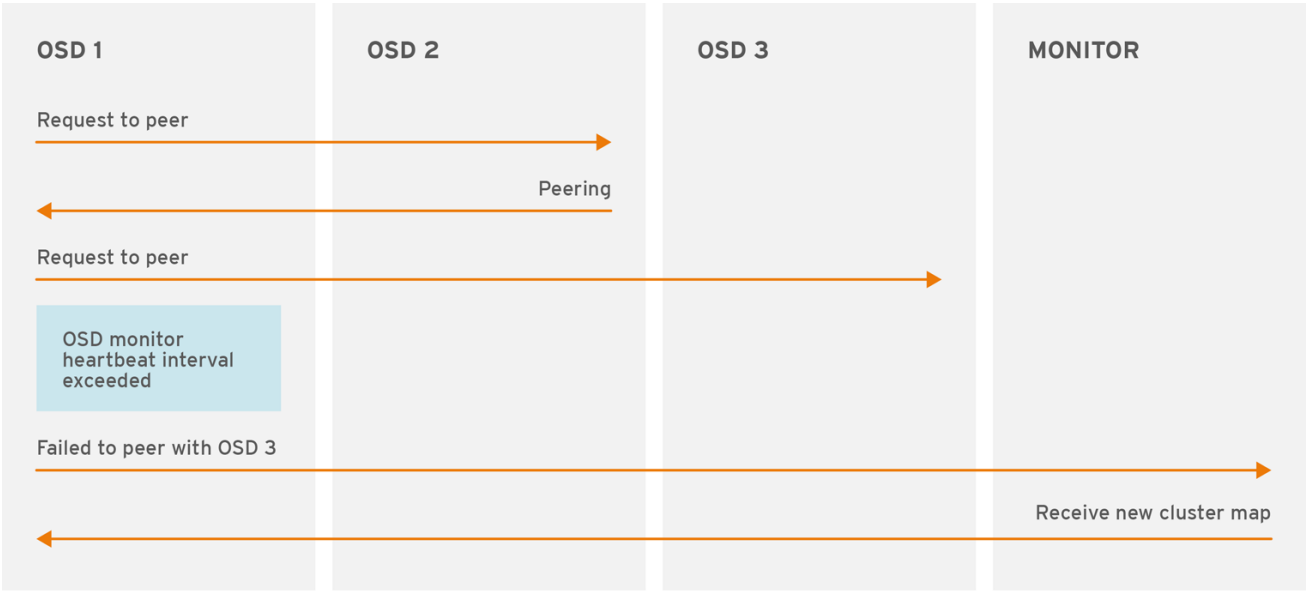
「誤報」を避けるために、Ceph は障害を報告したピアを、同様に遅延している「サブクラスター」の代理として考えます。これは必ずしもそうとは限りませんが、管理者が、パフォーマンスの低下しているシステムのサブセットに局所的に適切な補正を適用するのに役立つ場合があります。

Ceph は **mon_osd_reporter_subtree_level** 設定を使用して、CRUSH マップの共通の先復元タイプでピアを「subcluster」にグループ化します。デフォルトでは、**異なるサブツリー**からわずか 2 つのレポートは、他の Ceph OSD デーモン **down** を報告する必要があります。管理者は、Ceph 設定ファイルの **[mon]** セクションの下で、**mon_osd_min_down_reporters** 設定および **mon_osd_reporter_subtree_level** 設定を追加するか、ランタイム時に値を設定することで、Ceph Monitor に Ceph OSD Daemon **down** を報告するために必要な固有のサブツリーと共通の祖先型からレポーターの数を変更することができます。



7.3. OSD レポートのピアリングの失敗

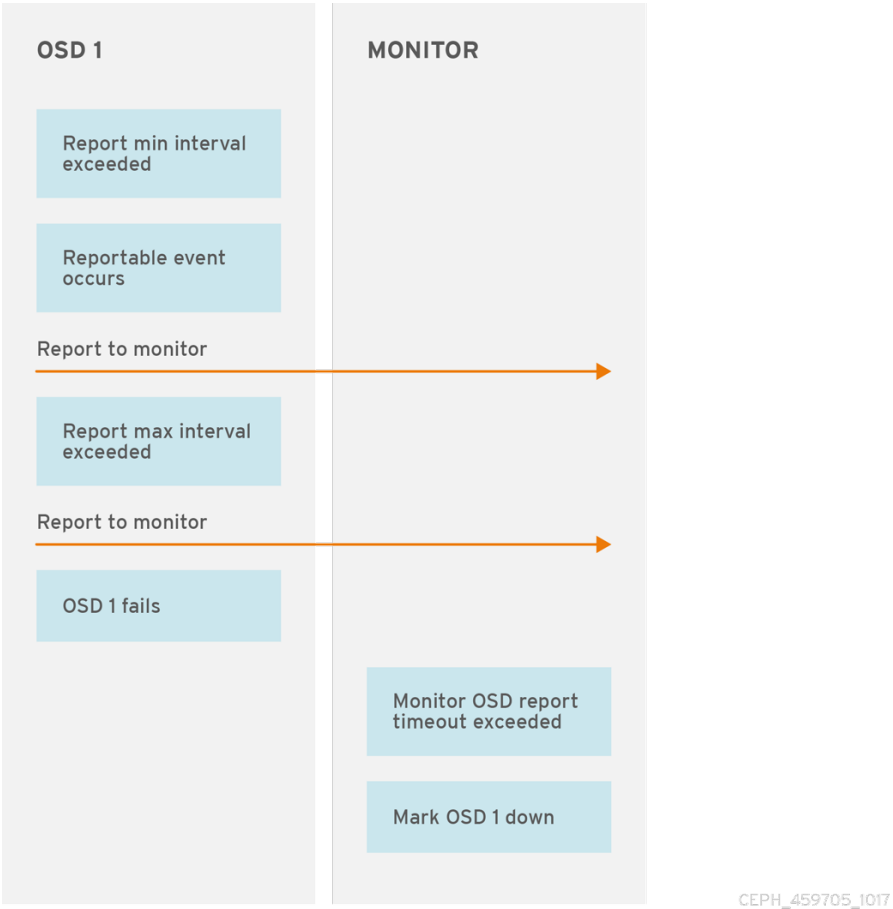
Ceph OSD デーモンが、その Ceph 設定ファイルまたはクラスターマップで定義された Ceph OSD デーモンのいずれともピアリングできない場合、30 秒ごとにクラスターマップの最新コピーを求めて Ceph Monitor に ping を実行します。Ceph Monitor ハートビートの間隔は、Ceph 設定ファイルの **[osd]** セクションに **osd mon heartbeat interval** 設定を追加するか、ランタイムに値を設定して変更できます。



7.4. OSD はステータスを報告します

Ceph OSD デーモンが Ceph Monitor に報告しない場合、Ceph Monitor は **mon osd report timeout** 後に **down** した Ceph OSD デーモンを考慮します。Ceph OSD デーモンは、障害、配置グループ統計の変更、**up_thru** の変更、または 5 秒以内にブートするなどの報告可能なイベント時に、Ceph Monitor にレポートを送信します。Ceph OSD Daemon の最小レポート間隔を変更するには、Ceph 設定ファイルの **[osd]** セクションに **osd mon report interval min** 設定を追加するか、ランタイムに値を設定します。

Ceph OSD デーモンは、目立った変更があったかどうかにかかわらず、120 秒ごとに Ceph Monitor にレポートを送信します。Ceph Monitor のレポート間隔を変更するには、Ceph 設定ファイルの **[osd]** セクションに **osd mon report interval max** 設定を追加するか、値をランタイムに設定します。



7.5. 設定方法

ハートビート設定を変更する際には、Ceph 設定ファイルの **[global]** セクションにその設定を含めます。

7.5.1. 監視設定

mon_osd_min_up_ratio

詳細

Ceph が Ceph OSD デーモンを **down** とマークする前に **up** となる Ceph OSD デーモンの最小比率。

型

double

デフォルト

.3

mon_osd_min_in_ratio

詳細

Ceph が Ceph OSD デーモンを **out** とマークを付ける前に **in** となる Ceph OSD デーモンの最小比率。

型

double

デフォルト

.3

mon_osd_laggy_halflife

詳細

laggy 予測の秒数が減ります。

型

整数

デフォルト

60*60

mon_osd_laggy_weight

詳細

laggy 予測の減少時の新しいサンプルの重み。

型

double

デフォルト

0.3

mon_osd_laggy_max_interval

詳細

ラグ推定値の **laggy_interval** の最大値 (秒単位)。モニターは適応アプローチを使用して特定の OSD の **laggy_interval** を評価します。この値は、その OSD の猶予時間を算出するために使用されます。

型

整数

デフォルト

300

mon_osd_adjust_heartbeat_grace

詳細

true に設定すると、Ceph は **laggy**推定値に基づいてスケーリングします。

型

ブール値

デフォルト

true

mon_osd_adjust_down_out_interval

詳細

true に設定すると、Ceph は **laggy** 推定値に基づいてスケーリングされます。

型

ブール値

デフォルト

true

mon_osd_auto_mark_in

詳細

Ceph は、Ceph OSD デーモンのブートを、Ceph Storage Cluster の **in** とマークします。

型

ブール値

デフォルト

false

mon_osd_auto_mark_auto_out_in

詳細

Ceph は、Ceph Storage クラスターから自動的に **out** とマーク付けされた Ceph OSD デーモンの起動が、クラスター内 **in** にあるとマークされます。

型

ブール値

デフォルト

true

mon_osd_auto_mark_new_in

詳細

Ceph は、新しい Ceph OSD デーモンのブートを Ceph Storage Cluster の **in** とマークします。

型

ブール値

デフォルト

true

mon_osd_down_out_interval

詳細

Ceph が Ceph OSD デーモンを **down** および **out** マークした後に応答しない場合には、Ceph が待機する秒数。

型

32 ビット整数

デフォルト

600

mon_osd_downout_subtree_limit

詳細

Ceph が自動的に **out** とマークアウトする最大の CRUSH ユニットタイプ。

型

文字列

デフォルト

rack

mon_osd_reporter_subtree_level

詳細

この設定は、報告する OSD の親 CRUSH ユニットタイプを定義します。OSD は、応答しないピアを見つけた場合、モニターに障害レポートを送信します。モニターは報告された OSD の数を **down** とマークし、猶予期間後に **out** になる可能性があります。

型

文字列

デフォルト

host

mon_osd_report_timeout

詳細

応答しない Ceph OSD デーモンが **down** するまでの猶予期間 (秒単位)。

型

32 ビット整数

デフォルト

900

mon_osd_min_down_reporters

詳細

down な Ceph OSD デーモンの報告に必要な Ceph OSD デーモンの最小数。

型

32 ビット整数

デフォルト

2

7.5.2. OSD の設定

osd_heartbeat_address

詳細

ハートビート用の Ceph OSD デーモンのネットワークアドレス

型

アドレス

デフォルト

ホストアドレス

osd_heartbeat_interval

詳細

Ceph OSD デーモンがピアに ping を実行する頻度 (秒単位)

型

32 ビット整数

デフォルト

6

osd_heartbeat_grace

詳細

Ceph OSD デーモンに Ceph Storage Cluster が **down** とみなすハートビートが表示されなかった場合の経過時間。

型

32 ビット整数

デフォルト

20

osd_mon_heartbeat_interval

詳細

Ceph OSD デーモンピアがない場合に、Ceph OSD デーモンがCeph Monitor に ping を実行する頻度

型

32 ビット整数

デフォルト

30

osd_mon_report_interval_max

詳細

Ceph OSD デーモンが Ceph Monitor に報告しなければならなくなるまでに待機できる最大時間 (秒)

型

32 ビット整数

デフォルト

120

osd_mon_report_interval_min

詳細

Ceph OSD デーモンが起動またはその他の報告可能なイベントから Ceph Monitor に報告するまでに待機する最小秒数

型

32 ビット整数

デフォルト

5

有効な範囲

osd mon report interval max未満である必要があります。

osd_mon_ack_timeout

詳細

Ceph Monitor が統計情報の要求を確認するまでの待ち時間 (秒数)

型

32 ビット整数

デフォルト

30

第8章 ファイルストア設定リファレンス

8.1. 拡張属性

拡張属性 (XATTR) は、CephFS 設定の重要な側面です。一部のファイルシステムでは、拡張属性に格納されるバイト数に制限があります。さらに、場合によっては、ファイルシステムが拡張属性を保存する代替方法ほど高速ではない可能性があります。以下の設定は、基盤となるファイルシステムに固有の拡張属性を格納する方法を使用することで、CephFS のパフォーマンスを向上させます。

Ceph 拡張属性は、サイズ制限を課さない場合、基盤となるファイルシステムによって提供される拡張属性を使用して、**inline xattr** として格納されます。サイズ制限がある場合 (たとえば、ext4 で合計 4KB)、**filestore max inline xattr size** または **filestore max inline xattrs** しきい値に達すると、一部の Ceph 拡張属性が **omap** と呼ばれるキーと値のデータベースに格納されます。

filestore_xattr_use_omap

説明

XATTRS のオブジェクトマップを使用します。ext4 ファイルシステムの場合は **true** に設定します。

タイプ

ブール値

必須

いいえ

デフォルト

false

filestore_omap_header_cache_size

説明

オブジェクト **omap** ヘッダーのキャッシュに使用される LRU のサイズを決定します。値が大きいほど、より多くのメモリーを使用しますが、**omap** でのルックアップを減らすことができます。(エキスパートのみ)。

タイプ

整数

デフォルト

1024

filestore_omap_backend

説明

omap に使用されるバックエンドを決定するために使用されます。**leveldb** または **rocksdb** に設定できます。（エクスパーとのみ。**rocksdb** は実験的)

タイプ

文字列

デフォルト

leveldb

filestore_debug_omap_check

説明

デバッグチェックは同期時にチェックされます。費用のかかる。デバッグ専用。

タイプ

ブール値

必須

いいえ

デフォルト

0

filestore_max_inline_xattr_size

説明

オブジェクトごとのファイルシステム (XFS、btrfs、ext4 など) に保存される拡張属性の最大サイズ。ファイルシステムが処理できるサイズよりも大きくしないでください。

タイプ

32 ビット符号なし整数

必須/任意

いいえ

デフォルト

512

filestore_max_inline_xattrs

説明

オブジェクトごとにファイルシステムに保存される拡張属性の最大数。

タイプ

32 ビット整数

必須

いいえ

デフォルト

2

filestore_max_inline_xattr_size_xfs

説明

オブジェクトごとの XFS ファイルシステムのファイルシステムに格納される拡張属性の最大サイズ。ファイルシステムが処理できるサイズよりも大きくしないでください。

タイプ

32 ビット符号なし整数

デフォルト

65536

filestore_max_inline_xattr_size_btrfs

説明

オブジェクトごとの btrfs のファイルシステムに保存される拡張属性の最大サイズ。ファイルシステムが処理できるサイズよりも大きくしないでください。

タイプ

32 ビット符号なし整数

デフォルト

2048

filestore_max_inline_xattr_size_other

説明

オブジェクトごとの btrfs または XFS 以外のファイルシステムのファイルシステムに保存される拡張属性の最大サイズ。ファイルシステムが処理できるサイズよりも大きくしないでください。

タイプ

32 ビット符号なし整数

デフォルト

512

filestore_max_inline_xattrs

説明

オブジェクトごとにファイルシステムに保存される拡張属性の最大数。きめの細かい設定をオーバーライドします。

タイプ

32 ビット符号なし整数

デフォルト

0

filestore_max_inline_xattrs_xfs

説明

オブジェクトごとに XFS ファイルシステムに保存される拡張属性の最大数。

タイプ

32 ビット符号なし整数

デフォルト

10

filestore_max_inline_xattrs_btrfs

説明

オブジェクトごとに btrfs ファイルシステムに保存される拡張属性の最大数。

タイプ

32 ビット符号なし整数

デフォルト

10

filestore_max_inline_xattrs_other

説明

オブジェクトあたりの btrfs または XFS 以外のファイルシステムに保存される拡張属性の最大数。

タイプ

32 ビット符号なし整数

デフォルト

2

8.2. 同期間隔

ファイルストアは定期的に書き込み操作を停止し、ファイルシステムを同期する必要があります。これにより、一貫したコミットポイントが作成されます。その後、ジャーナル項目をコミットポイントまで解放できます。より頻繁に同期すると、同期の実行に必要な時間が短縮され、ジャーナルに残す必要のあるデータの量が減る傾向があります。同期の頻度を減らすと、バッキングファイルシステムが小さな書き込みとメタデータの更新をより最適に結合できるようになり、より効率的な同期が可能になる可能性があります。

filestore_max_sync_interval

説明

ファイルストアを同期する最大間隔 (秒単位)。

タイプ

double

必須

いいえ

デフォルト

5

filestore_min_sync_interval

説明

ファイルストアを同期するための最小間隔 (秒単位)。

タイプ

double

必須

いいえ

デフォルト

.01

8.3. フラッシャー

ファイルストアフラッシャーは、最終的な同期のコストを削減するために、同期の前に **sync file range** オプションを使用して大規模な書き込み操作からのデータを強制的に書き出させます。実際には、ファイルストアフラッシャーを無効にすると、場合によってはパフォーマンスが向上するようです。

filestore_flusher

説明

ファイルストアフラッシャーを有効にします。

タイプ

ブール値

必須

いいえ

デフォルト

false

filestore_flusher_max_fds

説明

フラッシャのファイル記述子の最大数を設定します。

タイプ

整数

必須

いいえ

デフォルト

512

filestore_sync_flush

説明

同期フラッシャーを有効にします。

タイプ

ブール値

必須

いいえ

デフォルト

false

filestore_fsync_flushes_journal_data

説明

ファイルシステムの同期中にジャーナルデータをフラッシュします。

タイプ

ブール値

必須

いいえ

デフォルト

false

8.4. QUEUE

以下の設定は、ファイルストアキューのサイズの制限を提供します。

filestore_queue_max_ops

説明

新規操作をブロックする前にファイルストアが受け入れる実行中の操作の最大数を定義します。

タイプ

整数

必須/任意

いいえ。パフォーマンスへの影響を最小限に抑えます。

デフォルト

500

filestore_queue_max_bytes

説明

操作の最大バイト数。

タイプ

整数

必須

いいえ

デフォルト

100 << 20

filestore_queue_committing_max_ops

説明

ファイルストアがコミットできる操作の最大数。

タイプ

整数

必須

いいえ

デフォルト

500

filestore_queue_committing_max_bytes

説明

ファイルストアがコミットできる最大バイト数。

タイプ

整数

必須

いいえ

デフォルト

100 << 20

8.5. ライトバックスロットル

ページキャッシュはダーティデータを長く保持しすぎる傾向があるため、Ceph はカーネルのライトバック動作の一部を複製します。

filestore_wbthrottle_enable

説明

ファイルストアのライトバックスロットルを有効にします。ファイルストアのライトバックスロットルは、各ファイルストアの同期前にコミットされていない大量のデータが蓄積されるのを防ぐために使用されます。(エキスパートのみ)。

タイプ

ブール値

デフォルト

true

filestore_wbthrottle_btrfs_bytes_start_flusher

説明

Ceph が btrfs ファイルシステムのバックグラウンドフラッシュを開始するダーティーバイトしきい値。

タイプ

64 ビット未署名の整数

デフォルト

41943040

filestore_wbthrottle_btrfs_bytes_hard_limit

説明

フラッシャーが btrfs に追いつくまで、Ceph が I/O のスロットルを開始するダーティーバイトのしきい値。

タイプ

64 ビット未署名の整数

デフォルト

419430400

filestore_wbthrottle_btrfs_ios_start_flusher

説明

Ceph が btrfs のバックグラウンドフラッシュを開始するダーティ I/O しきい値。

タイプ

64 ビット未署名の整数

デフォルト

500

filestore_wbthrottle_btrfs_ios_hard_limit

説明

フラッシャーが btrfs に追いつくまで、Ceph が IO のスロットルを開始するダーティー I/O しきい値。

タイプ

64 ビット未署名の整数

デフォルト

5000

filestore_wbthrottle_btrfs_inodes_start_flusher

説明

Ceph が btrfs のバックグラウンドフラッシュを開始するダーティ inode しきい値。

タイプ

64 ビット未署名の整数

デフォルト

500

filestore_wbthrottle_btrfs_inodes_hard_limit

説明

フラッシャが btrfs に追いつくまで、Ceph が IO のスロットルを開始するダーティ inode しきい値。**fd** 制限未満である必要があります。

タイプ

64 ビット未署名の整数

デフォルト

5000

filestore_wbthrottle_xfs_bytes_start_flusher

説明

Ceph が XFS ファイルシステムのバックグラウンドフラッシュを開始するダーティーバイトのしきい値。

タイプ

64 ビット未署名の整数

デフォルト

41943040

filestore_wbthrottle_xfs_bytes_hard_limit

説明

フラッシャーが XFS に追いつくまで、Ceph が IO のスロットルを開始するダーティーバイトのしきい値。

タイプ

64 ビット未署名の整数

デフォルト

419430400

filestore_wbthrottle_xfs_ios_start_flusher

説明

Ceph が XFS のバックグラウンドフラッシュを開始するダーティ I/O しきい値。

タイプ

64 ビット未署名の整数

デフォルト

500

filestore_wbthrottle_xfs_ios_hard_limit

説明

フラッシャーが XFS に追いつくまで、Ceph が IO のスロットルを開始するダーティー I/O しきい値。

タイプ

64 ビット未署名の整数

デフォルト

5000

filestore_wbthrottle_xfs_inodes_start_flusher

説明

Ceph が XFS のバックグラウンドフラッシュを開始するダーティ inode しきい値。

タイプ

64 ビット未署名の整数

デフォルト

500

filestore_wbthrottle_xfs_inodes_hard_limit

説明

フラッシャが XFS に追いつくまで、Ceph が IO のスロットルを開始するダーティ inode しきい値。**fd** 制限未満である必要があります。

タイプ

64 ビット未署名の整数

デフォルト

5000

8.6. タイムアウト

filestore_op_threads

説明

並行して実行されるファイルシステム操作スレッドの数。

タイプ

整数

必須

いいえ

デフォルト

2

filestore_op_thread_timeout

説明

ファイルシステム操作スレッドのタイムアウト (秒単位)。

タイプ

整数

必須

いいえ

デフォルト

60

filestore_op_thread_suicide_timeout

説明

コミットをキャンセルする前のコミット操作のタイムアウト (秒単位)。

タイプ

整数

必須

いいえ

デフォルト

180

8.7. B ツリーファイルシステム

filestore_btrfs_snap

説明

btrfs ファイルストアのスナップショットを有効にします。

タイプ

ブール値

必須/任意

いいえ。btrfs にのみ使用されます。

デフォルト

true

filestore_btrfs_clone_range

説明

btrfs ファイルストアのクローン範囲を有効にします。

タイプ

ブール値

必須/任意

いいえ。btrfs にのみ使用されます。

デフォルト

true

8.8. JOURNAL

filestore_journal_parallel

説明

btrfs のデフォルトである並列ジャーナリングを有効にします。

タイプ

ブール値

必須

いいえ

デフォルト

false

filestore_journal_writeahead

説明

先行書き込みジャーナリングを有効にします。XFS のデフォルトです。

タイプ

ブール値

必須

いいえ

デフォルト

false

filestore_journal_trailing

説明

非推奨です。使用しないでください。

タイプ

ブール値

必須

いいえ

デフォルト

false

8.9. その他

filestore_merge_threshold

説明

親にマージする前のサブディレクトリー内のファイルの最小数 注: 負の値は、サブディレクトリーのマージを無効にすることを意味します。

タイプ

整数

必須

いいえ

デフォルト

10

filestore_split_multiple

説明

filestore_split_multiple * abs (filestore_merge_threshold) * 16 は、子ディレクトリーに分割される前のサブディレクトリー内のファイルの最大数です。

タイプ

整数

必須

いいえ

デフォルト

2

filestore_update_to

説明

ファイルストアの自動アップグレードを指定したバージョンに制限します。

タイプ

整数

必須

いいえ

デフォルト

1000

filestore_blackhole

説明

フロアに新しいトランザクションをドロップします。

タイプ

ブール値

必須

いいえ

デフォルト

false

filestore_dump_file

説明

ストアトランザクションがダンプされるファイル。

タイプ

ブール値

必須

いいえ

デフォルト

false

filestore_kill_at

説明

n 番目の機会に失敗を注入します。

タイプ

文字列

必須

いいえ

デフォルト

false

filestore_fail_eio

説明

EIO で予期せず失敗または終了します。

タイプ

ブール値

必須

いいえ

デフォルト

true

第9章 ジャーナル設定リファレンス

Ceph OSD は、以下の理由によりジャーナルを使用します。

速度

ジャーナルにより、Ceph OSD デーモンは小規模な書き込み操作を迅速にコミットできます。Ceph は小さなランダム I/O をジャーナルに順次書き込みます。これにより、バッキングファイルシステムが書き込み操作を結合する時間を増やすことができるため、バーストワークロードが高速化される傾向があります。ただし、Ceph OSD Daemon のジャーナルは、ファイルシステムがジャーナルに追いつくため、高速書き込みの短いスパートの後に、書き込みの進行がない期間が続くというスパイクパフォーマンスにつながる可能性があります。

一貫性

Ceph OSD デーモンには、アトミックな複合操作を保証するファイルシステムインターフェイスが必要です。Ceph OSD デーモンは、操作の説明をジャーナルに書き込み、その操作をファイルシステムに適用します。これにより、オブジェクト (配置グループのメタデータなど) へのアトミックな更新が可能になります。ファイルストアの **filestore max sync interval** および **filestore min sync interval** の設定の間で数秒ごとに、Ceph OSD は書き込み操作を停止し、ジャーナルをファイルシステムと同期します。これにより、Ceph OSD はジャーナルから操作をトリミングし、スペースを再利用できます。失敗すると、Ceph OSD は、最後の同期操作の後に開始するジャーナルを再生します。

9.1. 設定

Ceph OSD デーモンは、以下のジャーナル設定をサポートします。

journal_dio

説明

ジャーナルへのダイレクト I/O を有効にします。**journal block align** オプションを **true** に設定する必要があります。

タイプ

ブール値

必須/任意

aio を使用する場合は yes

デフォルト

true

journal_aio

説明

ジャーナルへの非同期書き込みに **libaio** の使用を有効にします。**journal dio** オプションを **true** に設定する必要があります。

タイプ

ブール値

必須/任意

いいえ

デフォルト

true.

journal_block_align

説明

ブロックは書き込み操作に合わせます。**dio** および **aio** には必須です。

タイプ

ブール値

必須/任意

dio および **aio** を使用する場合は yes

デフォルト

true

journal_max_write_bytes

説明

ジャーナルが一度に書き込む最大バイト数。

タイプ

整数

必須

いいえ

デフォルト

10 << 20

journal_max_write_entries

説明

ジャーナルが一度に書き込むエントリーの最大数。

タイプ

整数

必須

いいえ

デフォルト

100

journal_queue_max_ops

説明

キューで一度に許可される操作の最大数。

タイプ

整数

必須

いいえ

デフォルト

500

journal_queue_max_bytes

説明

キューで一度に許可される最大バイト数。

タイプ

整数

必須

いいえ

デフォルト

10 << 20

journal_align_min_size

説明

指定された最小値より大きいデータペイロードを整列します。

タイプ

整数

必須

いいえ

デフォルト

64 << 10

journal_zero_on_create

説明

ファイルストアが、**0's during `mkfs** で、ジャーナル全体を上書きするようになります。

タイプ

ブール値

必須

いいえ

デフォルト

false

第10章 ロギング設定リファレンス

Ceph 設定ファイルでロギングおよびデバッグの設定は必要ありませんが、必要に応じてデフォルト設定を上書きできます。

このオプションは、チャネルに関係なく、すべてのデーモンのデフォルトであると仮定される単一の項目を取ります。たとえば、「info」の指定は「default=info」と解釈されます。ただし、オプションはキーと値のペアを取ることもできます。たとえば、「default=daemon audit=local0」は『「すべてのデーモンのデフォルトで」「audit」を「local0」で上書きする』と解釈されます。

Ceph では、以下の設定がサポートされます。

log_file

詳細

クラスターのロギングファイルの場所

型

文字列

必須

いいえ

デフォルト

/var/log/ceph/$cluster-$name.log

mon_cluster_log_file

詳細

モニタークラスターのログファイルの場所

型

文字列

必須

いいえ

デフォルト

/var/log/ceph/$cluster.log

log_max_new

詳細

新規ログファイルの最大数

型

整数

必須

いいえ

デフォルト

1000

log_max_recent

詳細

ログファイルに追加する最近のイベントの最大数

型

整数

必須

いいえ

デフォルト

1000000

log_flush_on_exit

詳細

終了後に Ceph がログファイルをフラッシュするかどうかを決定します。

型

ブール値

必須

いいえ

デフォルト

true

mon_cluster_log_file_level

詳細

モニタークラスターのファイルロギングのレベル。有効な設定には、"debug"、"info"、"sec"、"warn"、および "error" が含まれます。

型

文字列

デフォルト

"info"

log_to_stderr

詳細

ロギングメッセージが標準エラー (**stderr**) で表示されるかどうかを確認します。

型

ブール値

必須

いいえ

デフォルト

true

err_to_stderr

詳細

エラーメッセージが標準エラー (**stderr**) で表示されるかどうかを確認します。

型

ブール値

必須

いいえ

デフォルト

true

log_to_syslog

詳細

ロギングメッセージが **syslog** に表示されるかどうかを決定します。

型

ブール値

必須

いいえ

デフォルト

false

err_to_syslog

詳細

エラーメッセージが **syslog** に表示されるかどうかを確認します。

型

ブール値

必須

いいえ

デフォルト

false

clog_to_syslog

詳細

clog メッセージが **syslog** に送信されるかどうかを決定します。

型

ブール値

必須

いいえ

デフォルト

false

mon_cluster_log_to_syslog

詳細

クラスターログが **syslog** に出力されるかどうかを確認します。

型

ブール値

必須

いいえ

デフォルト

false

mon_cluster_log_to_syslog_level

詳細

モニタークラスターの syslog ロギングのレベル。有効な設定には、"debug"、"info"、"sec"、"warn"、および "error" が含まれます。

型

文字列

デフォルト

"info"

mon_cluster_log_to_syslog_facility

詳細

syslog 出力を生成するファシリティー。通常、これは Ceph デーモンの「daemon」に設定されます。

型

文字列

デフォルト

"daemon"

clog_to_monitors

詳細

clog メッセージをモニターに送信するかどうかを決定します。

型

ブール値

必須

いいえ

デフォルト

true

mon_cluster_log_to_graylog

詳細

クラスターがログメッセージを graylog に出力するかどうかを決定します。

型

文字列

デフォルト

"false"

mon_cluster_log_to_graylog_host

詳細

graylog ホストの IP アドレス。graylog ホストがモニターホストと異なる場合は、適切な IP アドレスでこの設定を上書きします。

型

文字列

デフォルト

"127.0.0.1"

mon_cluster_log_to_graylog_port

詳細

graylog ログは、このポートに送信されます。データの受信用にポートが開いていることを確認します。

型

文字列

デフォルト

"12201"

10.1. OSD

osd_preserve_trimmed_log

詳細

トリミング後にトリミングされたログを保持します。

型

ブール値

必須

いいえ

デフォルト

false

osd_tmapput_sets_uses_tmap

詳細

tmap を使用します。デバッグ用途のみ。

型

ブール値

必須

いいえ

デフォルト

false

osd_min_pg_log_entries

詳細

配置グループのログエントリーの最小数

型

32 ビット未署名の整数

必須

いいえ

デフォルト

1000

osd_op_log_threshold

詳細

1つのパスで表示する op ログメッセージの数

型

整数

必須

いいえ

デフォルト

5

10.2. ファイルストア

filestore_debug_omap_check

説明

デバッグチェックは同期時にチェックされます。これはコストのかかる操作です。

タイプ

ブール値

必須

いいえ

デフォルト

0

10.3. CEPH OBJECT GATEWAY

rgw_log_nonexistent_bucket

説明

存在しないバケットをログに記録します。

タイプ

ブール値

必須

いいえ

デフォルト

false

rgw_log_object_name

説明

オブジェクトの名前をログに記録します。

タイプ

文字列

必須

いいえ

デフォルト

%Y-%m-%d-%H-%i-%n

rgw_log_object_name_utc

説明

オブジェクトログ名には UTC が含まれます。

タイプ

ブール値

必須

いいえ

デフォルト

false

rgw_enable_ops_log

説明

すべての RGW 操作のロギングを有効にします。

タイプ

ブール値

必須

いいえ

デフォルト

true

rgw_enable_usage_log

説明

RGW の帯域幅使用量のロギングを有効にします。

タイプ

ブール値

必須

いいえ

デフォルト

true

rgw_usage_log_flush_threshold

説明

保留中のログデータをフラッシュするしきい値。

タイプ

整数

必須

いいえ

デフォルト

1024

rgw_usage_log_tick_interval

説明

保留中のログデータを毎 **s** 秒ごとにフラッシュします。

タイプ

整数

必須

いいえ

デフォルト

30

rgw_intent_log_object_name

説明, タイプ

文字列

必須

いいえ

デフォルト

%Y-%m-%d-%i-%n

rgw_intent_log_object_name utc

説明

intent ログオブジェクト名に UTC タイムスタンプを含めます。

タイプ

ブール値

必須

いいえ

デフォルト

false