Red Hat Ceph Storage 7

Object Gateway ガイド

Ceph Object Gateway のデプロイ、設定および管理

Last Updated: 2024-04-25

Red Hat Ceph Storage 7 Object Gateway ガイド

Ceph Object Gateway のデプロイ、設定および管理

法律上の通知

概要

本書では、Ceph Object Gateway 環境のデプロイ、設定、および管理に関するガイダンスを提供します。本書では、読者が論理的な進捗パスをたどれるように Day Zero、Day One および Day Two で編成する方法を使用します。Day Zero は、潜在的なソリューションを実装する前に調査と計画が行われる場所です。第 1 章と第 2 章を参照してください。Day One は、ソフトウェアの実際のデプロイメントとインストールが実行される段階です。第 3 章を参照してください。Day Two は、すべての基本的な設定および詳細設定が行われる段階です。第 4 章および第 5 章、および第 6 章を参照してください。 Red Hat では、コード、ドキュメント、Web プロパティーにおける配慮に欠ける用語の置き換えに取り組んでいます。まずは、マスター (master)、スレーブ (slave)、ブラック

リスト (blacklist)、ホワイトリスト (whitelist) の 4 つの用語の置き換えから始めます。この取り組みは膨大な作業を要するため、今後の複数のリリースで段階的に用語の置き換えを実施して参ります。詳細は、Red Hat CTO である Chris Wright のメッセージ を参照してください。

目次

第1章 CEPH OBJECT GATEWAY

Ceph Object Gateway は RADOS Gateway (RGW) としても知られている、**librados** ライブラリー上に構築されたオブジェクトストレージインターフェイスであり、アプリケーションに Ceph ストレージクラスターへの RESTful ゲートウェイを提供します。Ceph Object Gateway は以下の 3 つのインターフェイスをサポートします。

S3-compatibility:

Amazon S3 RESTful API の大規模なサブセットと互換性のあるインターフェイスでオブジェクトストレージ機能を提供します。

S3 select を実行して、スループットを加速できます。ユーザーは、メディエーターなしで S3 select クエリーを直接実行できます。CSV 用と Apache Parquet (Parquet) 用の 2 つの S3 選択ワークフローがあり、CSV および Parquet オブジェクトを使用した S3 選択操作を提供します。これらの S3 選択操作の詳細は、**Red Hat Ceph Storage 開発者ガイド** のセクション **S3 選択操作** を参照してください。

Swift-compatibility:

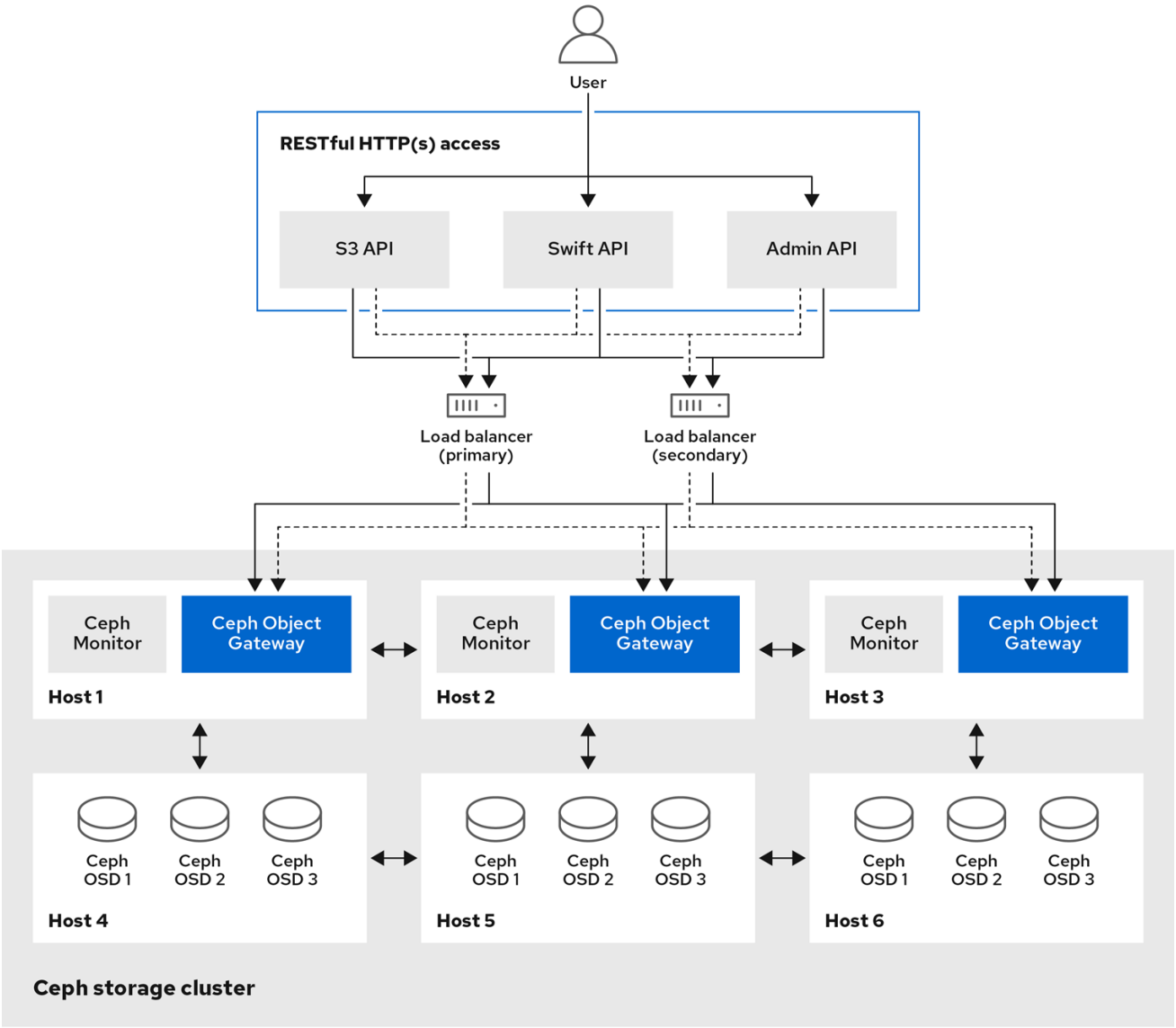
OpenStack Swift API の大規模なサブセットと互換性のあるインターフェイスでオブジェクトストレージ機能を提供します。

Ceph Object Gateway は、Ceph ストレージクラスターと対話するサービスです。OpenStack Swift および Amazon S3 と互換性のあるインターフェイスを提供するため、Ceph Object Gateway には独自のユーザー管理システムがあります。Ceph Object Gateway は、Ceph ブロックデバイスクライアントからのデータを保存するために使用される同じ Ceph ストレージクラスターにデータを保存できますが、これには別個のプールが使用され、別の CRUSH 階層も使用される可能性があります。S3 と Swift API は共通の namespace を共有するため、ある API でデータを作成してから、これを別の API で取得することができます。

管理用 API:

Ceph Object Gateway を管理するための管理インターフェイスを提供します。

管理 API 要求は、**admin** リソースのエンドポイントで始まる URI で行われます。管理 API の認可は S3 認可メカニズムを複製します。一部の操作では、ユーザーに特別な管理機能が必要です。応答タイプは、要求で format オプションを指定することにより、XML または JSON のいずれかになりますが、デフォルトでは JSON 形式になります。



WORM の概要

Write-Once-Read-Many (WORM) は、本番ゾーンでオブジェクトやバケットが侵害された場合でもデータ保護とデータ取得を保証するために使用される安全なデータストレージモデルです。

Red Hat Ceph Storage では、Write-Once-Read-Many (WORM) モデルを使用してオブジェクトとバケットを格納し、削除や上書きを防止するために、読み取り専用機能を備えた S3 Object Lock を使用することでデータセキュリティーを実現します。Red Hat Ceph Storage 管理者であっても削除することはできません。

S3 オブジェクトロックには 2 つの保持モードが用意されています。

- GOVERNANCE
- COMPLIANCE

これらの保持モードは、オブジェクトに対して異なる保護レベルを適用します。どちらの保持モードも、オブジェクトロックによって保護されているオブジェクトバージョンに適用できます。

GOVERNANCE モードでは、特別なパーミッションがない限り、ユーザーはオブジェクトバージョンの上書きや削除、あるいはロック設定の変更を行うことはできません。GOVERNANCE モードを使用する

と、ほとんどのユーザーによる削除からオブジェクトを保護できますが、必要に応じて一部のユーザーに保持設定を変更したり、オブジェクトを削除したりする権限を付与することもできます。

COMPLIANCE モードでは、どのユーザーも保護されたオブジェクトのバージョンを上書きしたり削除したりすることはできません。オブジェクトが COMPLIANCE モードでロックされている場合、その保持モードを変更したり短縮したりすることはできません。

関連情報

- 詳細は **Red Hat Ceph Storage Object Gateway ガイド** の **S3 のオブジェクトロックの有効化** を参照してください。

第2章 考慮事項および推奨事項

ストレージ管理者として、Ceph Object Gateway を実行してマルチサイトの Ceph Object Gateway ソリューションを実装する前に考慮すべき点について基本的に理解していることが重要です。ここでは、ハードウェアおよびネットワークの要件、Ceph Object Gateway で適切に機能するワークロードの種類、および Red Hat の推奨事項を把握することができます。

前提条件

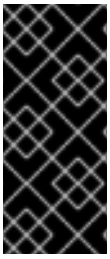
- ストレージソリューションを理解、検討、計画する時間を確保する。

2.1. RED HAT CEPH STORAGE のネットワークに関する考察

クラウドストレージソリューションの重要な点は、ネットワークのレイテンシーなどの要因により、ストレージクラスターが IOPS 不足になることです。また、ストレージクラスターがストレージ容量を使い果たす、はるか前に、帯域幅の制約が原因でスループットが不足することがあります。つまり、価格対性能の要求を満たすには、ネットワークのハードウェア設定が選択されたワークロードをサポートする必要があります。

ストレージ管理者は、ストレージクラスターをできるだけ早く復旧することを望みます。ストレージクラスターネットワークの帯域幅要件を慎重に検討し、ネットワークリンクのオーバーサブスクリプションに注意してください。また、クライアント間のトラフィックからクラスター内のトラフィックを分離します。また、SSD (Solid State Disk) やフラッシュ、NVMe などの高性能なストレージデバイスの使用を検討する場合には、ネットワークパフォーマンスの重要性が増していることも考慮してください。

Ceph はパブリックネットワークとストレージクラスターネットワークをサポートしています。パブリックネットワークは、クライアントのトラフィックと Ceph Monitor との通信を処理します。ストレージクラスターネットワークは、Ceph OSD のハートビート、レプリケーション、バックフィル、リカバリーのトラフィックを処理します。ストレージハードウェアには、**最低でも** 10GB のイーサネットリンクを 1 つ使用し、接続性とスループット向けにさらに 10GB イーサネットリンクを追加できます。



重要

Red Hat では、レプリケートされたプールをもとに **osd_pool_default_size** を使用してパブリックネットワークの倍数となるように、ストレージクラスターネットワークに帯域幅を割り当てることを推奨しています。また、Red Hat はパブリックネットワークとストレージクラスターネットワークを別々のネットワークカードで実行することを推奨しています。

重要

Red Hat では、実稼働環境での Red Hat Ceph Storage のデプロイメントに 10GB のイーサネットを使用することを推奨しています。1GB のイーサネットネットワークは、実稼働環境のストレージクラスターには適していません。

ドライブに障害が発生した場合、1 GB イーサネットネットワーク全体で 1 TB のデータをレプリケートするには 3 時間かかります。3 TB には 9 時間かかります。3TB を使用するのが一般的なドライブ設定です。一方、10GB のイーサネットネットワークの場合、レプリケーションにかかる時間はそれぞれ 20 分、1 時間となります。Ceph OSD が失敗すると、ストレージクラスターは、障害のある OSD と同じ障害ドメインおよびデバイスクラスに含まれるデータをレプリケートして復元することに注意してください。

ラックなどの大規模なドメインに障害が発生した場合は、ストレージクラスターが帯域幅を大幅に消費します。複数のラックで設定されるストレージクラスター (大規模なストレージ実装では一般的) を構築

する際には、最適なパフォーマンスを得るために、ファットツリー設計でスイッチ間のネットワーク帯域幅をできるだけ多く利用することを検討してください。一般的な 10 GB のイーサネットスイッチには、48 個の 10 GB ポートと 4 個の 40 GB のポートがあります。スループットを最大にするには、Spine (背骨) で 40 GB ポートを使用します。または、QSFP+ および SFP+ ケーブルを使用する未使用の 10 GB ポートを別のラックおよびスパインルーターに接続するために、さらに 40 GB のポートに集計することを検討します。また、LACP モード 4 でネットワークインターフェイスを結合することも検討してください。また、特にバックエンドやクラスターのネットワークでは、ジャンボフレーム、最大伝送単位 (MTU) 9000 を使用してください。

Red Hat Ceph Storage クラスターをインストールしてテストする前に、ネットワークのスループットを確認します。Ceph のパフォーマンスに関する問題のほとんどは、ネットワークの問題から始まります。Cat-6 ケーブルのねじれや曲がりといった単純なネットワークの問題は、帯域幅の低下につながります。フロント側のネットワークには、最低でも 10 GB のイーサネットを使用してください。大規模なクラスターの場合には、バックエンドやクラスターのネットワークに 40GB のイーサネットを使用することを検討してください。

重要

ネットワークの最適化には、CPU/帯域幅の比率を高めるためにジャンボフレームを使用し、非ブロックのネットワークスイッチのバックプレーンを使用することを Red Hat は推奨します。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、環境内のすべてのホスト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。

2.2. RED HAT CEPH STORAGE の基本的な考慮事項

Red Hat Ceph Storage を使用するための最初の考慮事項は、データのストレージストラテジーの開発についてです。ストレージストラテジーとは、特定のユースケースに対応するためのデータを保管する手法を指します。OpenStack などのクラウドプラットフォームのボリュームおよびイメージを保存する必要がある場合は、ジャーナル用に Solid State Drives(SSD) を使用する高速な Serial Attached SCSI(SAS) ドライブにデータを保存することができます。一方、S3 または Swift 準拠のゲートウェイのオブジェクトデータを保存する必要がある場合は、従来の Serial Advanced Technology Attachment(SATA) ドライブなど、より経済的な方法を使用できます。Red Hat Ceph Storage は、同じストレージクラスターの両方のシナリオに対応しますが、クラウドプラットフォーム用に高速ストレージストラテジーと、オブジェクトストア用に従来のストレージを提供する手段が必要です。

Ceph のデプロイメントを正常に実行するための最も重要な手順の 1 つとして、クラスターのユースケースとワークロードに適した価格性能比のプロファイルを特定します。ユースケースに適したハードウェアを選択することが重要です。たとえば、コールドストレージアプリケーション用に IOPS が最適化されたハードウェアを選択すると、ハードウェアのコストが必要以上に増加します。また、IOPS が重視されるワークロードにおいて、より魅力的な価格帯に対して容量が最適化されたハードウェアを選択すると、パフォーマンスの低下に不満を持つユーザーが出てくる可能性が高くなります。

Red Hat Ceph Storage は、複数のストレージストラテジーをサポートできます。健全なストレージ戦略を策定するには、ユースケース、費用対効果、パフォーマンスのトレードオフ、データの耐久性などを考慮する必要があります。

ユースケース

Ceph は大容量のストレージを提供し、多くのユースケースをサポートします。

- Ceph Block Device クライアントは、クラウドプラットフォーム向けの代表的なストレージバックエンドで、ボリュームやイメージに対して制限なくストレージを提供し、コピーオンライトクローニングなど、高パフォーマンス機能を備えています。

- Ceph Object Gateway クライアントは、音声、ビットマップ、ビデオなどのオブジェクト向けの RESTful S3 準拠のオブジェクトおよび Swift 準拠のオブジェクトストレージを提供するクラウドプラットフォームの主要なストレージバックエンドです。

- 従来のファイルストレージである Ceph ファイルシステム。

コスト vs. パフォーマンス

速度、サイズ、耐久性など高いほうが優れています。ただし、優れた品質にはそれぞれコストがかかるので、費用対効果の面でトレードオフがあります。パフォーマンスの観点からでは、以下のユースケースを考慮してください。SSD は、比較的小規模なデータおよびジャーナリングのために非常に高速ストレージを提供できます。データベースやオブジェクトインデックスの保存には、非常に高速な SSD のプールが有効ですが、他のデータの保存にはコストがかかりすぎてしまいます。SSD ジャーナリングのある SAS ドライブは、ボリュームやイメージを安価かつ高速なパフォーマンスで提供できます。SSD ジャーナリングのない SATA ドライブは、全体的なパフォーマンスは低くなりますが、ストレージの価格を安価に抑えることができます。OSD の CRUSH 階層を作成する場合は、ユースケースと許容コスト/パフォーマンスのトレードオフを考慮する必要があります。

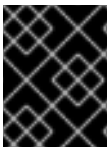
データの持続性

大規模なクラスターでは、ハードウェア障害は想定されており、例外ではありません。ただし依然として、データの損失および中断は受け入れられません。そのため、データの持続性は非常に重要になります。Ceph は、オブジェクトの複数のレプリカコピー、またはイレイジャーコーディングおよび複数のコーディングのチャンクでデータの持続性に対応します。複数のコピーまたはコーディングチャンクにより、さらに費用対効果の面でのトレードオフが分かります。コピーやコーディングのチャンクが少ない場合にはコストがかかりませんが、パフォーマンスが低下した状態で、書き込み要求に対応できなくなる可能性があります。通常、追加のコピーまたはコーディングチャンクが 2 つあるオブジェクトを使用すると、ストレージクラスターが復旧する間に、パフォーマンスが低下した状態でクラスターの書き込みを行うことができます。

レプリケーションでは、ハードウェア障害に備えて、障害ドメインをまたいで 1 つ以上のデータの冗長コピーを保存します。しかし、データの冗長コピーは、規模が大きくなるとコスト高になります。たとえば、1 ペタバイトのデータを 3 つのレプリケーションで保存するには、少なくとも容量が 3 ペタバイトあるストレージクラスターが必要になります。

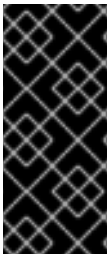
イレイジャーコーディングでは、データをデータチャンクとコーディングチャンクに分けて保存します。データチャンクが失われた場合には、イレイジャーコーディングにより、残りのデータチャンクとコーディングチャンクで失われたデータチャンクを回復できます。イレイジャーコーディングはレプリケーションに比べて大幅に経済的です。たとえば、データチャンク 8 つとコーディングチャンク 3 つのイレイジャーコーディングを使用すると、データのコピーが 3 つある状態と同じ冗長性が得られます。ただし、このようなエンコーディングスキームでは、初期のデータ保存量が約 1.5 倍になるのに対し、レプリケーションでは 3 倍になります。

CRUSH アルゴリズムは、Ceph が、ストレージクラスター内の異なる場所に追加のコピーまたはコーディングチャンクを保存して、このプロセスをサポートします。これにより、1 つのストレージデバイスまたはホストに障害が発生しても、データ損失を回避するために必要なコピーやコーディングチャンクがすべて失われないようにします。費用対効果の面でのトレードオフやデータの耐性を考慮してストレージ戦略を計画し、ストレージプールとして Ceph クライアントに提示します。



重要

データストレージプールのみがイレイジャーコーディングを使用できます。サービスデータやバケットインデックスを格納するプールはレプリケーションを使用します。



重要

Ceph のオブジェクトコピーやコーディングチャンクを使用すると、RAID ソリューションが古く感じられます。Ceph はすでにデータの持続性に対応しており、質の低い RAID ではパフォーマンスに悪影響があり、RAID を使用してデータを復元すると、ディープコピーや消失訂正を使用するよりもはるかにスピードが遅くなるので、RAID は使用しないでください。

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 の最小ハードウェア要件 セクションを参照してください。

2.2.1. Ceph デーモンの共存とその利点

コンテナー化された Ceph デーモンを同じホストの同じ場所に配置できます。Ceph のデーモンの一部を共存する利点を以下に示します。

- 小規模での総所有コスト (TCO) を大幅に改善します。
- 全体的なパフォーマンスを向上させることができます。
- 最小設定の物理ホストの量を減らします。
- リソースの使用率が向上します。
- Red Hat Ceph Storage のアップグレードが容易です。

コンテナーを使用すると、以下のリストの 1 つのデーモンを Ceph OSD デーモン (**ceph-osd**) と同じ場所に配置できます。さらに、Ceph Object Gateway (**radosgw**)、Ceph Metadata Server (**ceph-mds**)、および Grafana の場合は、Ceph OSD デーモンに加えて、以下のリストのデーモンと併置できます。

- Ceph メタデータサーバー (**ceph-mds**)
- Ceph Monitor (**ceph-mon**)
- Ceph Manager (**ceph-mgr**)
- NFS Ganesha (**nfs-ganesha**)
- Ceph マネージャー (**ceph-grafana**)

表2.1 デーモン配置の例

ホスト名	デーモン	デーモン	デーモン
host1	OSD	Monitor および Manager	Prometheus
host2	OSD	Monitor および Manager	RGW
host3	OSD	Monitor および Manager	RGW
host4	OSD	メタデータサーバー	

ホスト名	デーモン	デーモン	デーモン
host5	OSD	メタデータサーバー	

注記

ceph-mon と **ceph-mgr** は密接に連携するため、コロケーションの観点では 2 つの別のデーモンとはみなされません。

Ceph デーモンを共存させるには、コマンドラインインターフェイスから **ceph orch** コマンドに **--placement** オプションを指定するか、サービス仕様 YAML ファイルを使用することができます。

コマンドラインの例

```
[ceph: root@host01 /]# ceph orch apply mon --placement="host1 host2 host3"
```

サービス仕様の YAML ファイルの例

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

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

Red Hat は、Ceph Object Gateway を Ceph OSD コンテナーと併置してパフォーマンスを向上することを推奨します。追加のハードウェアコストを発生せずに最高のパフォーマンスを実現するには、ホストごとに 2 つの Ceph Object Gateway デーモンを使用します。

Ceph Object Gateway のコマンドラインの例

```
[ceph: root@host01 /]# ceph orch apply rgw example --placement="6 host1 host2 host3"
```

Ceph Object Gateway サービス仕様の YAML ファイルの例

```
service_type: rgw
service_id: example
placement:
  count: 6
  hosts:
   - host01
   - host02
   - host03
```

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

以下のダイアグラムは、同じ場所に置かれたデーモンと、同じ場所に置かれていないデーモンを使用するストレージクラスターの相違点を示しています。

図2.1 同じ場所に配置されたデーモン

図2.2 同じ場所に配置されていないデーモン

関連情報

- **--placement** オプションの使用に関する詳細は、**Red Hat Ceph Storage オペレーションガイド** の Ceph Orchestrator を使用したサービスの管理 の章を参照してください。
- 詳細は、Red Hat Ceph Storage RGW deployment strategies and sizing guidance のアーティクルを参照してください。

2.3. RED HAT CEPH STORAGE ワークロードに関する考慮事項

Ceph Storage クラスターの主な利点の 1 つとして、パフォーマンスドメインを使用して、同じストレージクラスター内のさまざまなタイプのワークロードをサポートする機能があります。各パフォーマンスドメインには、異なるハードウェア設定を関連付けることができます。ストレージ管理者は、ストレージプールを適切なパフォーマンスドメインに配置し、特定のパフォーマンスとコストプロファイル

に合わせたストレージをアプリケーションに提供できます。これらのパフォーマンスドメインに適切なサイズ設定と最適化されたサーバーを選択することは、Red Hat Ceph Storage クラスターを設計するのに不可欠な要素です。

データの読み取りおよび書き込みを行う Ceph クライアントインターフェイスに対して、Ceph Storage クラスターはクライアントがデータを格納する単純なプールとして表示されます。ただし、ストレージクラスターは、クライアントインターフェイスから完全に透過的な方法で多くの複雑な操作を実行します。Ceph クライアントおよび Ceph オブジェクトストレージデーモン (Ceph OSD または単に OSD) はいずれも、オブジェクトのストレージおよび取得にスケーラブルなハッシュ (CRUSH) アルゴリズムで制御されたレプリケーションを使用します。Ceph OSD は、ストレージクラスター内のコンテナーで実行できます。

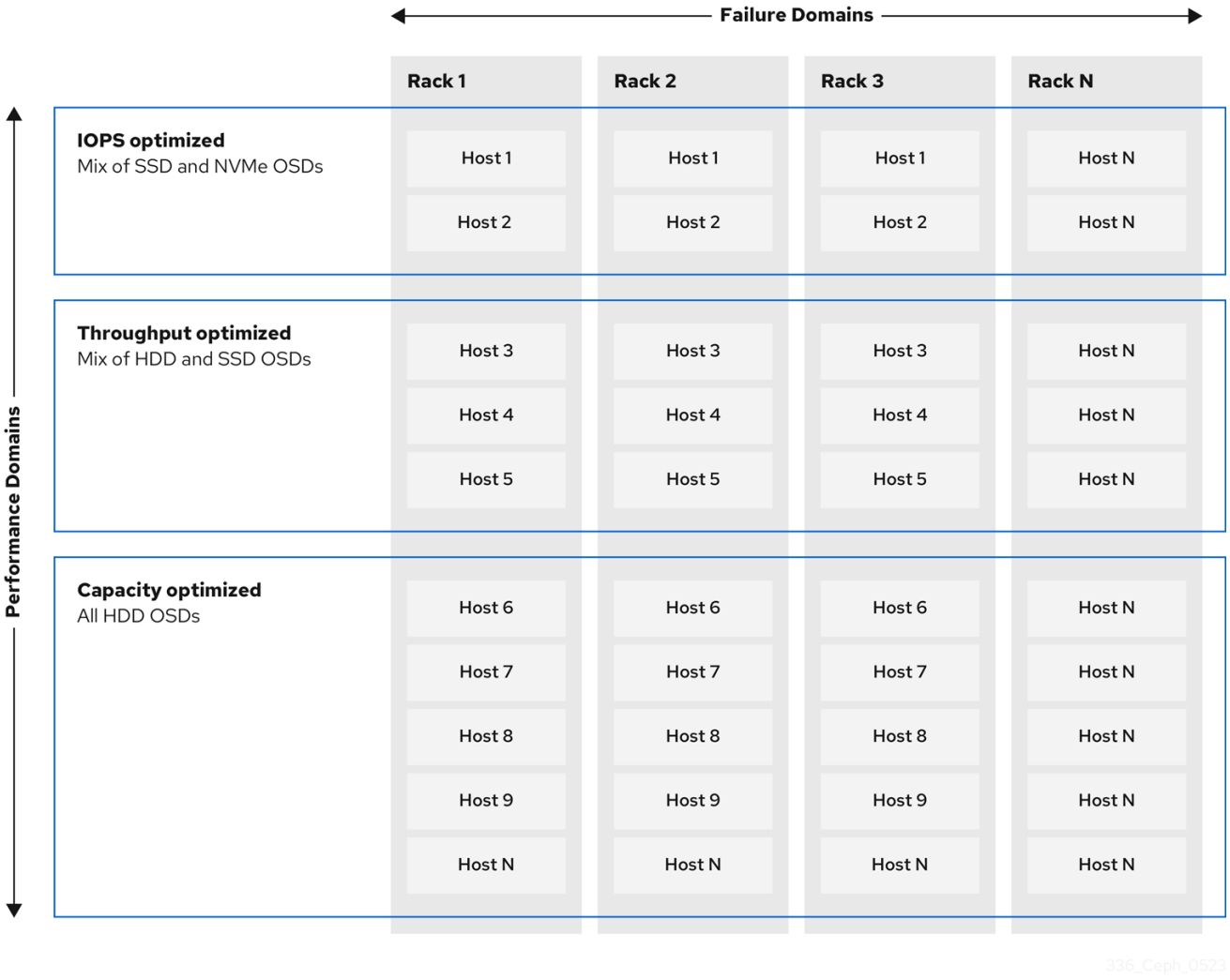
CRUSH マップはクラスターリソースのトポロジーを表し、マップは、クラスター内のクライアントホストと Ceph Monitor ホストの両方に存在します。Ceph クライアントおよび Ceph OSD はどちらも CRUSH マップと CRUSH アルゴリズムを使用します。Ceph クライアントは OSD と直接通信することで、オブジェクト検索の集中化とパフォーマンスのボトルネックとなる可能性を排除します。CRUSH マップとピアとの通信を認識することで、OSD は動的障害復旧のレプリケーション、バックフィル、およびリカバリーを処理できます。

Ceph は CRUSH マップを使用して障害ドメインを実装します。Ceph は CRUSH マップを使用してパフォーマンスドメインの実装も行います。パフォーマンスドメインは、基礎となるハードウェアのパフォーマンスプロファイルを反映させます。CRUSH マップは Ceph のデータの格納方法を記述し、これは単純な階層 (例: 非周期グラフ) およびルールセットとして実装されます。CRUSH マップは複数の階層をサポートし、ハードウェアパフォーマンスプロファイルのタイプを別のタイプから分離できます。Ceph では、デバイスの classes でパフォーマンスドメインを実装しています。

たとえば、これらのパフォーマンスドメインを同じ Red Hat Ceph Storage クラスター内に共存させることができます。

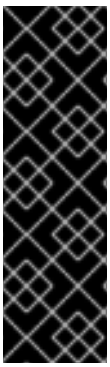
- ハードディスクドライブ (HDD) は、一般的にコストと容量を重視したワークロードに適しています。
- スループットを区別するワークロードは通常、ソリッドステートドライブ (SSD) の Ceph 書き込みジャーナルで HDD を使用します。
- MySQL や MariaDB のような IOPS を多用するワークロードでは、SSD を使用することが多いです。

図2.3 パフォーマンスおよび障害ドメイン



ワークロード

Red Hat Ceph Storage は、3 つの主要なワークロードに対して最適化されています。



重要

ストレージクラスターの価格とパフォーマンスに大きな影響を与えるので、どのハードウェアを購入するかを検討する**前**に、Red Hat Ceph Storage クラスターで実行するワークロードを慎重に検討してください。たとえば、ワークロードの容量が最適化されいるにも拘らず、スループットが最適化されたワークロードに、対象のハードウェアがより適している場合に、ハードウェアが必要以上に高価になってしまいます。逆に、ワークロードのスループットが最適化されていて、容量が最適化されたワークロードに、対象のハードウェアが適している場合は、ストレージクラスターのパフォーマンスが低下します。

- **IOPS を最適化:** IOPS (Input, Output per Second) が最適化されたデプロイメントは、MYSQL や MariaDB インスタンスを OpenStack 上の仮想マシンとして稼働させるなど、クラウドコンピューティングの操作に適しています。IOPS が最適化された導入では、15k RPM の SAS ドライブや、頻繁な書き込み操作を処理するための個別の SSD ジャーナルなど、より高性能なストレージが必要となります。一部の IOPS のシナリオでは、すべてのフラッシュストレージを使用して IOPS と総スループットが向上します。
  IOPS が最適化されたストレージクラスターには、以下のプロパティーがあります。

- IOPS あたり最小コスト
- 1 GB あたりの最大 IOPS。
- 99 パーセンタイルのレイテンシーの一貫性。

IOPS に最適化されたストレージクラスターの用途は以下のとおりです。

- 典型的なブロックストレージ。
- ハードドライブ (HDD) の 3x レプリケーションまたはソリッドステートドライブ (SSD) の 2x レプリケーション。
- OpenStack クラウド上の MySQL

- **最適化されたスループット**: スループットが最適化されたデプロイメントは、グラフィック、音声、ビデオコンテンツなどの大量のデータを提供するのに適しています。スループットが最適化されたデプロイメントには、高帯域幅のネットワークハードウェア、コントローラー、高速シーケンシャル読み取り/書き込み機能のあるハードディスクドライブが必要です。高速なデータアクセスが必要な場合は、スループットを最適化したストレージ戦略を使用します。また、高速な書き込み性能が必要な場合は、ジャーナルに SSD (Solid State Disks) を使用すると、書き込み性能が大幅に向上します。
スループットが最適化されたストレージクラスターには、以下のような特性があります。
  - MBps あたりの最小コスト (スループット)。
  - TB あたり最も高い MBps。
  - BTU あたりの最大 MBps
  - Watt あたりの MBps の最大数。
  - 97 パーセンタイルのレイテンシーの一貫性。

  スループットを最適化したストレージクラスターの用途は以下のとおりです。

  - ブロックまたはオブジェクトストレージ。
  - 3x レプリケーション。
  - ビデオ、音声、およびイメージのアクティブなパフォーマンスストレージ。
  - 4K 映像などのストリーミングメディア

- **最適化された容量**: 容量が最適化されたデプロイメントは、大量のデータを可能な限り安価に保存するのに適しています。容量が最適化されたデプロイメントは通常、パフォーマンスがより魅力的な価格と引き換えになります。たとえば、容量を最適化したデプロイメントでは、ジャーナリングに SSD を使用するのではなく、より低速で安価な SATA ドライブを使用し、ジャーナルを同じ場所に配置することがよくあります。
コストと容量が最適化されたストレージクラスターには、次のような特性があります。
  - TB あたり最小コスト
  - TB あたり最小の BTU 数。
  - TB あたりに必要な最小 Watt。

  コストと容量が最適化されたストレージクラスターの用途は以下のとおりです。

- 典型的なオブジェクトストレージ。
- 使用可能な容量を最大化するイレイジャーコーディング
- オブジェクトアーカイブ。
- ビデオ、音声、およびイメージオブジェクトのリポジトリー。

2.4. CEPH OBJECT GATEWAY の考慮事項

ストレージクラスターの設計に関するもう 1 つの重要な点として、ストレージクラスターが 1 つのデータセンターサイトにあるか、複数のデータセンターサイトにまたがるかどうかを判別することです。マルチサイトストレージクラスターは、地理的に分散したフェイルオーバーと、長期的な停電、地震、ハリケーン、洪水、その他の災害からの復旧の恩恵を受けます。さらに、マルチサイトストレージクラスターは active-active 設定を持つことができ、クライアントアプリケーションを最も近い利用可能なストレージクラスターに転送できます。これは、コンテンツ配信ネットワークに適したストレージストラテジーです。データをクライアントのできるだけ近くに配置することを検討してください。これは、ストリーミング 4k ビデオなど、スループット集約型のワークロードに重要です。

重要

Red Hat は、Ceph のストレージプールを作成する前に、レルム、ゾーングループ、およびゾーン名を特定することが推奨されます。一部のプール名の先頭に、ゾーン名を標準の命名規則として追加します。

関連情報

- 詳細は、**Red Hat Ceph Storage Object Gateway ガイド**の **マルチサイトの設定および管理** セクションを参照してください。

2.4.1. 管理データストレージ

Ceph Object Gateway は、インスタンスのゾーン設定で定義された一連のプールに管理データを保存します。たとえば、後続のセクションで説明したバケット、ユーザー、ユーザークォータおよび使用状況の統計は、Ceph Storage Cluster のプールに保存されます。デフォルトでは、Ceph Object Gateway は以下のプールを作成し、それらをデフォルトゾーンにマッピングします。

- **.rgw.root**
- **.default.rgw.control**
- **.default.rgw.meta**
- **.default.rgw.log**
- **.default.rgw.buckets.index**
- **.default.rgw.buckets.data**
- **.default.rgw.buckets.non-ec**

注記

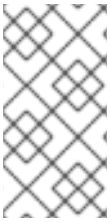
.default.rgw.buckets.index プールは、Ceph Object Gateway でバケットが作成された後にのみ作成されます。一方、データはバケットにアップロードされた後に **.default.rgw.buckets.data** プールが作成されます。

CRUSH ルールセットと配置グループの数を設定することができるように、これらのプールを手動で作成することを検討してください。一般的な設定では、Ceph Object Gateway の管理データを格納するプールは、管理データに 10 個のプールがあるため、多くの場合、同じ CRUSH ルールセットを使用し、使用する配置グループの数を少なくします。

Red Hat は、**.rgw.root** プールとサービスプールは同じ CRUSH 階層を使用し、CRUSH ルールの障害ドメインとして少なくとも **node** を使用することを推奨しています。Red Hat では、データの耐久性には **replicated** を使用し、**.rgw.root** プールには **erasure** を使用せず、サービスプールを使用することを推奨します。

mon_pg_warn_max_per_osd 設定は、プールに過剰な配置グループを割り当てると警告します (つまりデフォルトでは **300**)。この値は、ニーズやハードウェアの能力に合わせて調整することができ、**n** は OSD あたりの PG の最大数です。

```
mon_pg_warn_max_per_osd = n
```



注記

.rgw.root を含むサービスプールの場合は、Ceph placement groups (PGs) per pool calculator から提案される PG 数は、Ceph OSD あたりのターゲットの PG よりもはるかに少なくなります。また、Ceph OSD の数が、calculator のステップ 4 で設定されていることを確認します。

重要

ガベッジコレクションは、OMAP ではなく通常の RADOS オブジェクトで **.log** プールを使用します。今後のリリースでは、より多くの機能はメタデータを **.log** プールに保管します。したがって、Red Hat は、**.log** プールに NVMe/SSD Ceph OSD を使用することを推奨しています。

.RGW.root プール

Ceph Object Gateway 設定が保存されるプール。これには、レルム、ゾーングループ、およびゾーンが含まれます。通常、その名前はゾーン名の前に追加されません。

サービスプール

サービスプールは、サービス制御、ガベージコレクション、ロギング、ユーザー情報、および使用方法に関連するオブジェクトを保存します。慣例により、これらのプール名には、プール名の前にゾーン名が付加されます。

- **.ZONE_NAME.rgw.control**: コントロールプール。
- **.ZONE_NAME.log**: ログプールには、すべてのバケット/コンテナーのログおよび create、read、update、および delete などのオブジェクトアクションが含まれます。
- **.ZONE_NAME.rgw.buckets.index**: このプールはバケットのインデックスを保存します。
- **.ZONE_NAME.rgw.buckets.data**: このプールはバケットのデータを格納します。

- **.ZONE_NAME.rgw.meta**: メタデータプールは user_keys およびその他の重要なメタデータを保存します。
- **.ZONE_NAME.meta:users.uid**: ユーザー ID プールには、一意のユーザー ID のマップが含まれます。
- **.ZONE_NAME.meta:users.keys**: キープールには、各ユーザー ID のアクセスキーおよびシークレットキーが含まれます。
- **.ZONE_NAME.meta:users.email**: メールプールには、ユーザー ID に関連付けられたメールアドレスが含まれます。
- **.ZONE_NAME.meta:users.swift**: Swift プールには、ユーザー ID の Swift サブユーザー情報が含まれます。

関連情報

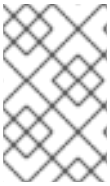
- 詳細は、**Red Hat Ceph Storage Object Gateway ガイド**の **プールについて** セクションを参照してください。
- 詳細は、**Red Hat Ceph Storage ストラテジーガイド**を参照してください。

2.4.2. インデックスプール

Ceph Object Gateway で使用する OSD ハードウェアを選択する場合 (ユースケースに関係なく)、インデックスプールを保存するために、SSD または NVMe ドライブのいずれかの高パフォーマンスドライブが 1 つ以上ある OSD ノードが必要です。これは、バケットに多数のオブジェクトが含まれる場合は、特に重要になります。

Red Hat Ceph Storage が Bluestore を実行している場合には、NVMe ドライブを別のプールではなく **block.db** デバイスとしてデプロイすることを推奨します。

Ceph Object Gateway インデックスデータはオブジェクトマップ (OMAP) にのみ書き込まれます。BlueStore の OMAP データは、OSD の **block.db** デバイスにあります。NVMe ドライブが HDD OSD の **block.db** デバイスとして機能し、インデックスプールが HDD OSD によって保持されている場合、インデックスデータは **block.db** デバイスにのみ書き込まれます。**block.db** パーティション/lvm がブロックの 4% に正しくサイズ設定されている限り、BlueStore ではこの設定だけで十分です。



注記

Red Hat は、インデックスプールの HDD デバイスに対応していません。サポート対象設定の情報については、**Red Hat Ceph Storage: Supported configurations**の記事を参照してください。

インデックスエントリーは約 200 バイトのデータで、**rocksdb** に OMAP として保存されます。これはごくわずかな量のデータですが、Ceph Object Gateway を使用すると、1 つのバケットに数千万から数億のオブジェクトが含まれる可能性があります。インデックスプールを高性能ストレージメディアの CRUSH 階層にマッピングすることにより、バケットに非常に多くのオブジェクトが含まれている場合に、レイテンシーが短くなり、パフォーマンスが劇的に向上します。

重要

実稼働クラスターでは、標準の OSD ノードには OSD ジャーナルを保存するための SSD または NVMe ドライブが 1 つ以上と、インデックスプールまたは **block.db** デバイスが含まれます。同じ物理ドライブには、別のパーティションまたは論理ボリュームを使用します。

2.4.3. データプール

データプールは、Ceph Object Gateway が特定のストレージポリシーのオブジェクトデータを保管する場所です。データプールは、サービスプールの PG 数を減らすのではなく、配置グループ (PG) を完全に補完します。データプールのイレイジャーコーディングの使用を検討してください。これはレプリケーションよりも大幅に効率的であり、データの持続性を維持しつつ容量要件を大幅に減らすことができます。

イレイジャーコーディングを使用するには、イレイジャーコードプロファイルを作成します。詳細は、**Red Hat Ceph Storage ストレージ戦略ガイド**の **コードプロファイルの消去** セクションを参照してください。

重要

プールの作成後にプロファイルを変更できないため、正しいプロファイルを選択することが重要です。プロファイルを変更するには、別のプロファイルで新しいプールを作成し、オブジェクトを古いプールから新しいプールに移行する必要があります。

デフォルト設定は 2 つのデータチャンクと 1 つのエンコーディングチャンクです。つまり、1 つの OSD のみが失われる可能性があります。回復性が高い場合は、大量のデータおよびエンコーディングチャンクを考慮してください。たとえば、一部の大規模なシステムでは、8 データチャンクと 3 つのエンコーディングチャンクが使用され、データを失うことなく 3 つの OSD が失敗することが可能になります。

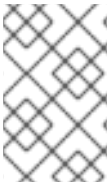
重要

各データおよびエンコードチャンク SHOULD は、最低でも別のノードまたはホストに保存されます。小規模なストレージクラスターの場合、これにより、大量のデータおよびエンコードチャンクを使用する場合に、**rack** の非現実障害ドメインを最低限の CRUSH 障害ドメインとして使用します。そのため、データプールは、最小 CRUSH 障害ドメインとして、**host** を持つ別の CRUSH 階層を使用するのが一般的です。Red Hat は、最小障害ドメインとして **host** を推奨します。イレイジャーコードチャンクが同じホスト内の Ceph OSD に保存されていると、ジャーナルやネットワークカードなどのホストの障害により、データが失われる可能性があります。

データの追加プールを作成するには、**ceph osd pool create** コマンドにプール名、PG および PGP の数、**erasure** データ永続性メソッド、イレイジャーコードプロファイル、およびルール名を指定して実行します。

2.4.4. データ追加プール

data_extra_pool は、イレイジャーコーディングを使用できないデータ向けです。たとえば、マルチパートアップロードにより、複数部分でのモジションなどの大規模なオブジェクトのアップロードが可能です。これらのパーツは、最初にイレイジャーコーディングなしで保存する必要があります。イレイジャーコーディングは、部分的なアップロードではなく、オブジェクト全体に適用されます。



注記

placement group (PG) per Pool Calculator では、**data_extra_pool** 対してプールあたりの PG 数を少なくすることが推奨されています。ただし、PG 数はサービスプールとバケットインデックスプールと同じ PG の約 2 倍です。

データの追加プールを作成するには、**ceph osd pool create** コマンドにプール名、PG および PGP の数、**replicated** データ永続性メソッド、およびルール名を指定して作成します。以下に例を示します。

```
# ceph osd pool create .us-west.rgw.buckets.non-ec 64 64 replicated rgw-service
```

2.5. CRUSH 階層の開発

ストレージ管理者は、Ceph Storage クラスターおよび Object Gateway のデプロイ時に、通常 Ceph Object Gateway にはデフォルトのゾーングループおよびゾーンがあります。Ceph ストレージクラスターにはデフォルトのプールがあり、次に、デフォルトの CRUSH 階層およびデフォルトの CRUSH ルールで CRUSH マップを使用します。

重要

デフォルトの **rbd** プールは、デフォルトの CRUSH ルールを使用できます。Ceph クライアントがデフォルトのルールまたは階層を使用してクライアントデータを保存している場合は、それらを削除 **しないでください**。

実稼働ゲートウェイは通常、ゲートウェイの使用および地理的な場所に従って名前が付けられるカスタムレルム、ゾーングループ、およびゾーンを使用します。また、Ceph ストレージクラスターには、複数の CRUSH 階層を持つ CRUSH マップがあります。

- **サービスプール:** 少なくとも 1 つの CRUSH 階層はサービスプール用であり、場合によってはデータ用になります。サービスプールには、**.rgw.root** と、ゾーンに関連付けられたサービスプールが含まれます。サービスプールは、通常単一の CRUSH 階層下にあり、データの持続性のためにレプリケーションを使用します。データプールは CRUSH 階層を使用することもできますが、通常プールはデータの耐久性のためにイレイジャーコーディングで設定されます。

- **インデックス:** 少なくとも 1 つの CRUSH 階層はインデックスプール用にある **必要があり**、CRUSH 階層は SSD ドライブや NVMe ドライブなどの高パフォーマンスのメディアにマップされます。バケットインデックスはパフォーマンスのボトルネックとなる可能性があります。Red Hat は、この CRUSH 階層で SSD または NVMe ドライブを使用することを推奨します。Ceph OSD ジャーナルに使用される SSD または NVMe ドライブのインデックス用にパーティションを作成します。さらに、インデックスはバケットシャーディングで設定する必要があります。

- **配置プール:** 各配置ターゲットの配置プールには、バケットインデックス、データバケット、およびバケットの追加が含まれます。これらのプールは、個別の CRUSH 階層下に分類できます。Ceph Object Gateway は複数のストレージポリシーをサポートすることができるため、ストレージポリシーのバケットプールは異なる CRUSH 階層に関連付け、IOPS 最適化、スループット最適化、容量最適化などの異なるユースケースを反映できます。バケットインデックスプールには、SSD ドライブ、NVMe ドライブなどの高性能記憶媒体にバケットインデックスプールをマップするために、独自の CRUSH 階層を使用 **すべきです**。

2.5.1. CRUSH ルートの作成

管理ノードのコマンドラインから、各 CRUSH 階層に対して CRUSH マップに CRUSH ルートを作成します。また、潜在的にデータストレージプールを担うことができるサービスプールに、少なくとも 1 つ

の CRUSH 階層がある **必要があります**。そのような SSD、NVMe ドライブなどの高性能ストレージメディアにマッピングされたバケットインデックスプールに、少なくとも 1 つの CRUSH 階層がある **はずです**。

CRUSH 階層の詳細は、**Red Hat Ceph Storage Storage ストラテジーガイド 7**の CRUSH Hierarchies セクションを参照してください。

CRUSH マップを手動で編集するには、**Red Hat Ceph Storage Storage ストラテジーガイド 7**の **CRUSH Map の編集** セクションを参照してください。

以下の例では、**data0**、**data1**、および **data2** という名前のホストは、同じ物理ホストを参照する CRUSH の階層が複数存在するため、**data0-sas-ssd**、**data0-index** などの拡張論理名を使用します。

一般的な CRUSH ルートは、SAS ドライブを持つノードとジャーナル用の SSD を表す可能性があります。以下に例を示します。

```
##
# SAS-SSD ROOT DECLARATION
##

root sas-ssd {
  id -1   # do not change unnecessarily
  # weight 0.000
  alg straw
  hash 0  # rjenkins1
  item data2-sas-ssd weight 4.000
  item data1-sas-ssd weight 4.000
  item data0-sas-ssd weight 4.000
}
```

バケットインデックスの CRUSH ルートは、SSD や NVMe ドライブなどの高パフォーマンスメディアを表す **はずです**。OSD ジャーナルを格納する SSD または NVMe メディアにパーティションを作成することを検討してください。以下に例を示します。

```
##
# INDEX ROOT DECLARATION
##

root index {
  id -2    # do not change unnecessarily
  # weight 0.000
  alg straw
  hash 0  # rjenkins1
  item data2-index weight 1.000
  item data1-index weight 1.000
  item data0-index weight 1.000
}
```

2.5.2. CRUSH マップでの論理ホスト名の使用

RHCS 3 以降のリリースでは、CRUSH はストレージデバイスの class の概念をサポートします。これは、RHCS 2 以前のバージョンではサポートされないためです。NVMe、SSD、HDD などの複数のストレージデバイスを含むホストまたはノードを持つ RHCS 3 クラスターでは、デバイスクラスを持つ単一の CRUSH 階層を使用して、ストレージデバイスの異なるクラスを区別します。これにより、論理ホス

ト名を使用する必要がなくなります。RHCS 2 以前のリリースでは、複数の CRUSH 階層を使用し、デバイスの各クラスに 1 つ、論理ホスト名を使用して CRUSH 階層内のホストまたはノードを区別します。

RHCS 3 以降のリリースでは、CRUSH はストレージデバイスの class の概念をサポートします。これは、RHCS 2 以前のバージョンではサポートされないためです。NVMe、SSD、HDD などの複数のストレージデバイスを含むホストまたはノードを持つ RHCS 3 クラスターでは、デバイスクラスを持つ単一の CRUSH 階層を使用して、ストレージデバイスの異なるクラスを区別します。これにより、論理ホスト名を使用する必要がなくなります。RHCS 2 以前のリリースでは、複数の CRUSH 階層を使用し、デバイスの各クラスに 1 つ、論理ホスト名を使用して CRUSH 階層内のホストまたはノードを区別します。

CRUSH マップでは、ホスト名は一意で、1 回のみ使用する必要があります。ホストが複数の CRUSH 階層とユースケースに対応する場合、CRUSH マップは実際のホスト名の代わりに論理ホスト名を使用して、ホスト名が一度だけ使用されるようにします。たとえば、ノードには、SSD などの複数のドライブ、SSD ジャーナルを備えた SAS ドライブ、および同一ジャーナルを持つ SATA ドライブが複数ある場合があります。RHCS 2 以前のリリースでは、同じホストに複数の CRUSH 階層を作成するには、階層は実際のホスト名の代わりに論理ホスト名を使用する必要があります。そのため、バケット名は CRUSH 階層内で一意となるようにします。たとえば、ホスト名が **data2** の場合、CRUSH 階層は、**data2-sas-ssd**、**data2-index** などの論理名を使用する可能性があります。

```
host data2-sas-ssd {
  id -11   # do not change unnecessarily
  # weight 0.000
  alg straw
  hash 0  # rjenkins1
  item osd.0 weight 1.000
  item osd.1 weight 1.000
  item osd.2 weight 1.000
  item osd.3 weight 1.000
}
```

前述の例では、ホスト **data2** は論理名 **data2-sas-ssd** を使用して、SSD にジャーナルのある SAS ドライブを 1 つの階層にマッピングします。**osd.0** から **osd.3** の OSD ID は、高スループットのハードウェア設定で SSD ジャーナルを使用する SAS ドライブを表しています。これらの OSD ID は、以下の例の OSD ID とは異なります。

以下の例では、ホスト **data2** は論理名 **data2-index** を使用してバケットインデックスの SSD ドライブを 2 つ目の階層にマッピングします。**osd.4** の OSD ID は、バケットインデックスプール専用に使用される SSD ドライブまたはその他の高速ストレージメディアを示しています。

```
host data2-index {
  id -21   # do not change unnecessarily
  # weight 0.000
  alg straw
  hash 0  # rjenkins1
  item osd.4 weight 1.000
}
```

重要

論理ホスト名を使用する場合は、以下の設定のいずれかが Ceph 設定ファイルに存在することを確認します。これにより、OSD 起動スクリプトが起動時に実際のホスト名を使用しなくなり、CRUSH マップのデータの検索に失敗します。

CRUSH マップが論理ホスト名を使用する場合、この例では、OSD の起動スクリプトが、初期化時に実際のホスト名に従ってホストを特定しないようにします。Ceph 設定ファイルの **[global]** セクションに、以下の設定を追加します。

```
osd_crush_update_on_start = false
```

論理ホスト名を定義する別の方法は、Ceph 設定ファイルの **[osd.<ID>]** セクションで各 OSD の CRUSH マップの場所を定義することです。これにより、OSD の起動スクリプトが定義する場所が上書きされます。前述の例からは、エントリーは以下のようになります。

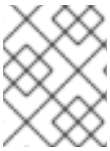
```
[osd.0]
osd crush location = "host=data2-sas-ssd"

[osd.1]
osd crush location = "host=data2-sas-ssd"

[osd.2]
osd crush location = "host=data2-sas-ssd"

[osd.3]
osd crush location = "host=data2-sas-ssd"

[osd.4]
osd crush location = "host=data2-index"
```

重要

CRUSH マップが実際のホスト名ではなく論理ホスト名を使用する場合、Ceph Storage Cluster は OSD が実際のホスト名にマップされ、実際のホスト名は CRUSH マップにないことを想定し、Ceph Storage Cluster は OSD が実際のホスト名にないことを想定し、Ceph Storage Cluster クライアントは OSD とそのデータを見つけません。

2.5.3. CRUSH ルールの作成

デフォルトの CRUSH 階層と同様に、CRUSH マップにはデフォルトの CRUSH ルールも含まれます。



注記

デフォルトの **rbd** プールはこのルールを使用できます。他のプールを使用して顧客データを保存する場合は、デフォルトのルールを削除しないでください。

CRUSH ルールに関する一般的な情報は、Red Hat Ceph Storage 7 の**Red Hat Ceph Storage Storage ストラテジーガイド** の CRUSH ルール セクションを参照してください。CRUSH マップを手動で編集するには、Red Hat Ceph Storage 7 の **Red Hat Ceph Storage Storage ストラテジーガイド** で CRUSH map の編集 を参照してください。

CRUSH 階層ごとに、CRUSH ルールを作成します。以下の例は、**.rgw.root** を含むサービスプールを保存する CRUSH 階層のルールを示しています。この例では、ルート **sas-ssd** がメインの CRUSH 階層として機能します。デフォルトのルールと区別するために、**rgw-service** という名前を使用します。**step take sas-ssd** 行は、CRUSH ルートの作成 で作成された **sas-ssd** ルートを使用するようにプールに指示します。このルートの子バケットには、SAS ドライブを備えた OSD と、高スループットハードウェア設定のジャーナルに対して SSD または NVMe ドライブなどの高性能ストレージメディアが含まれます。**step chooseleaf** の **type rack** 部分が障害ドメインになります。以下の例では、ラックです。

-

```
##
# SERVICE RULE DECLARATION
##

rule rgw-service {
 type replicated
 min_size 1
 max_size 10
 step take sas-ssd
 step chooseleaf firstn 0 type rack
 step emit
}
```

注記

前述の例では、データが 3 回複製される 3 回複製される場合は、同様の数の OSD ノードを含むクラスターに 3 つ以上のラックが存在する必要があります。

ヒント

type replicated 設定は、データ永続性、レプリカ数、またはイレイジャーコーディングとは **関係ありません。複製** のみがサポートされます。

以下の例は、データプールを保存する CRUSH 階層のルールを示しています。この例では、root **sas-ssd** は、サービスルールと同じ CRUSH 階層としてメインの CRUSH 階層として機能します。 **rgw-throughput** を使用して、デフォルトのルールと **rgw-service** と区別します。**step take sas-ssd** 行は、CRUSH ルートの作成 で作成された **sas-ssd** ルートを使用するようにプールに指示します。このルートの子バケットには、SAS ドライブを備えた OSD と、高スループットハードウェア設定の SSD または NVMe ドライブなどの高性能ストレージメディアが含まれます。**step chooseleaf** の **type host** 部分障害ドメインになります。以下の例では、ホストです。ルールは同じ CRUSH 階層を使用し、異なる障害ドメインを使用することに注意してください。

```
##
# THROUGHPUT RULE DECLARATION
##

rule rgw-throughput {
 type replicated
 min_size 1
 max_size 10
 step take sas-ssd
 step chooseleaf firstn 0 type host
 step emit
}
```

注記

前述の例では、プールが大量のデータでイレイジャーコーディングを使用し、デフォルトよりもエンコーディングのチャンクを使用する場合、イレイジャーコーディングのチャンクを容易にするために、同様の数の OSD ノードを含むクラスター内のラックが少なくとも数ある必要があります。小規模なクラスターの場合、これは実用的ではない可能性があるため、前述の例では **host** を CRUSH 障害ドメインとして使用します。

以下の例は、インデックスプールを保存する CRUSH 階層のルールを示しています。この例では、root の **index** は主な CRUSH 階層として機能します。**rgw-index** を使用して、**rgw-service** と **rgw-throughput** と区別します。**step take index** 行は、CRUSH ルートの作成 で作成された **index** root を使用するようにプールに指示します。その子バケットには、SSD ドライブ、NVMe ドライブなどの高性能ストレージメディア、または OSD ジャーナルも格納する SSD ドライブまたは NVMe ドライブ上のパーティションが含まれます。**step chooseleaf** の **type rack** 部分が障害ドメインになります。以下の例では、ラックです。

```
##
# INDEX RULE DECLARATION
##

rule rgw-index {
 type replicated
 min_size 1
 max_size 10
 step take index
 step chooseleaf firstn 0 type rack
 step emit
}
```

関連情報

- CRUSH 階層に関する一般的な情報は、**Red Hat Ceph Storage ストレージストラテジーガイド** の CRUSH Administration セクションを参照してください。

2.6. CEPH OBJECT GATEWAY のマルチサイトに関する考慮事項

Ceph Object Gateway のマルチサイト設定には、少なくとも 2 つの Red Hat Ceph Storage クラスターが必要です。また、少なくとも 2 つの Ceph オブジェクトゲートウェイインスタンス (各 Red Hat Ceph Storage クラスターに 1 つずつ) が必要です。通常、2 つの Red Hat Ceph Storage クラスターは地理的に別々の場所に置かれますが、同じ物理サイトにある 2 つの Red Hat Ceph Storage クラスターで同じマルチサイトで作業することができます。

マルチサイト設定には、プライマリーゾーングループとプライマリーゾーンが必要です。また、各ゾーングループにはプライマリーゾーンが必要です。ゾーングループに 1 つ以上のセカンダリーゾーンが含まれる可能性があります。

注記

マルチサイトは、CLI または Red Hat Ceph Storage ダッシュボードから設定できます。詳細は、**Ceph ダッシュボードでのマルチサイトオブジェクトゲートウェイの設定** を参照してください。

重要

レルムのプライマリーゾーングループ内のプライマリーゾーンは、ユーザー、クォータ、バケットなどのレルムのメタデータのプライマリーコピーを保存します。このメタデータは、セカンダリーゾーンおよびセカンダリーゾーングループに自動的に同期されます。**radosgw-admin** コマンドラインインターフェイス (CLI) で発行されるメタデータ操作は、セカンダリーゾーングループおよびゾーンと確実に同期するために、プライマリーゾーングループのプライマリーゾーン内のノードで発行する **必要があります**。現在、セカンダリーゾーンとゾーングループでメタデータ操作を発行することは **可能です** が、同期 **されず**、メタデータが断片化されるため、推奨 **されません**。

以下の図は、マルチサイトの Ceph Object Gateway 環境で可能な 1 つおよび 2 つのレルム設定を示しています。

図2.4 1つのレルム

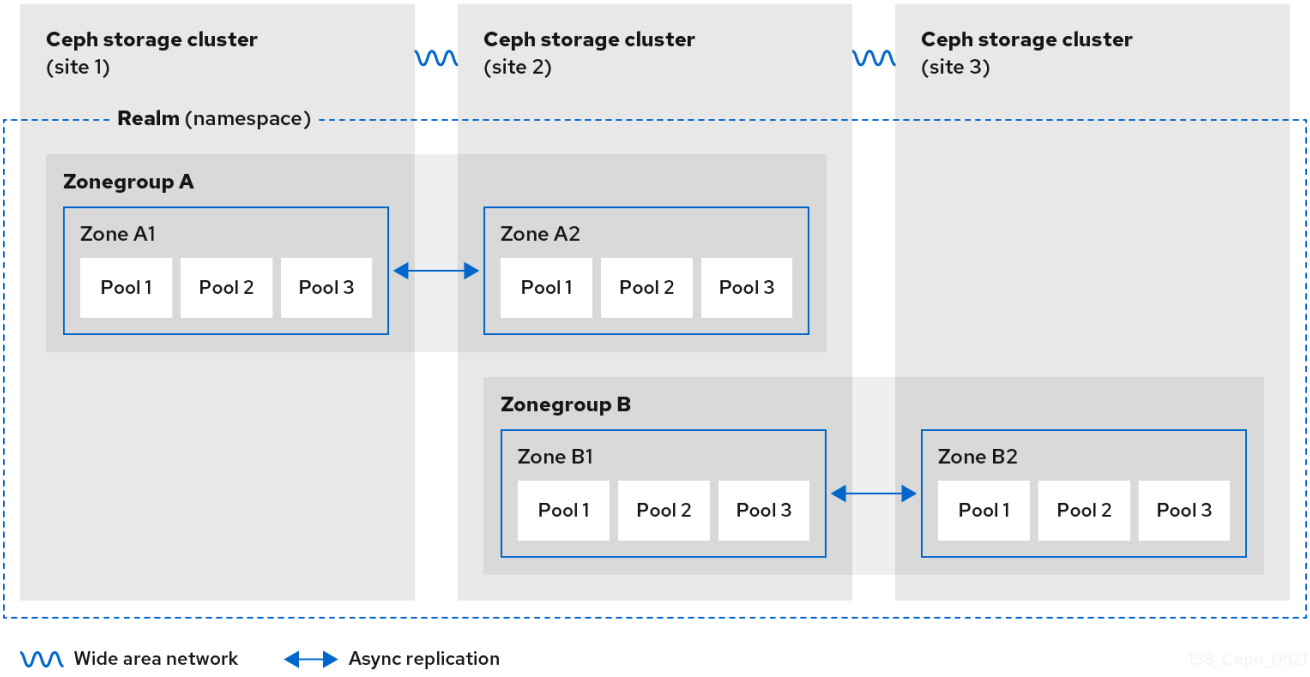


図2.5 2 つのレルム

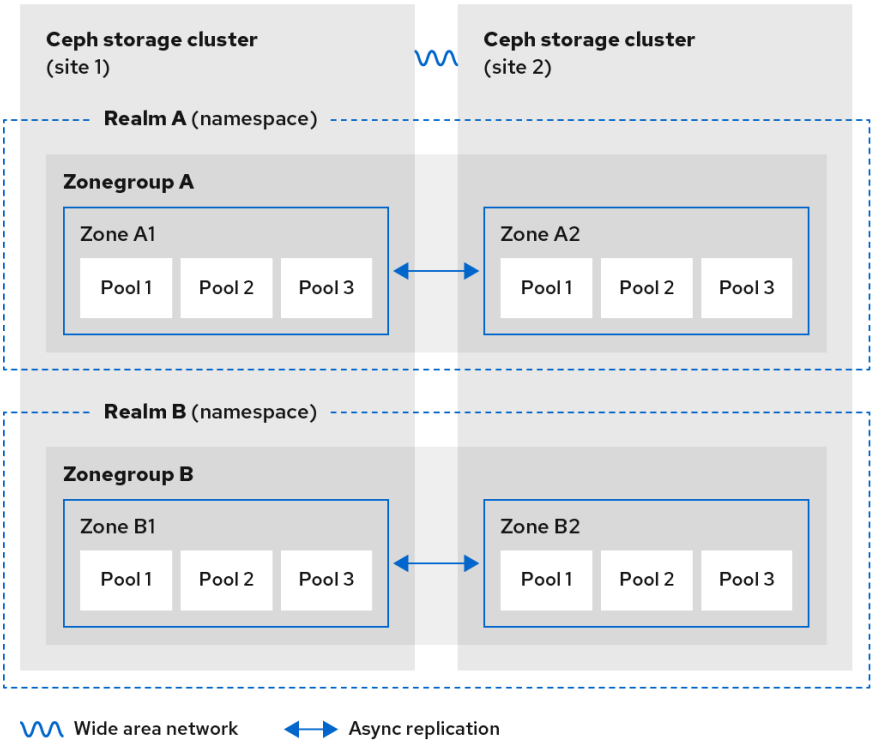
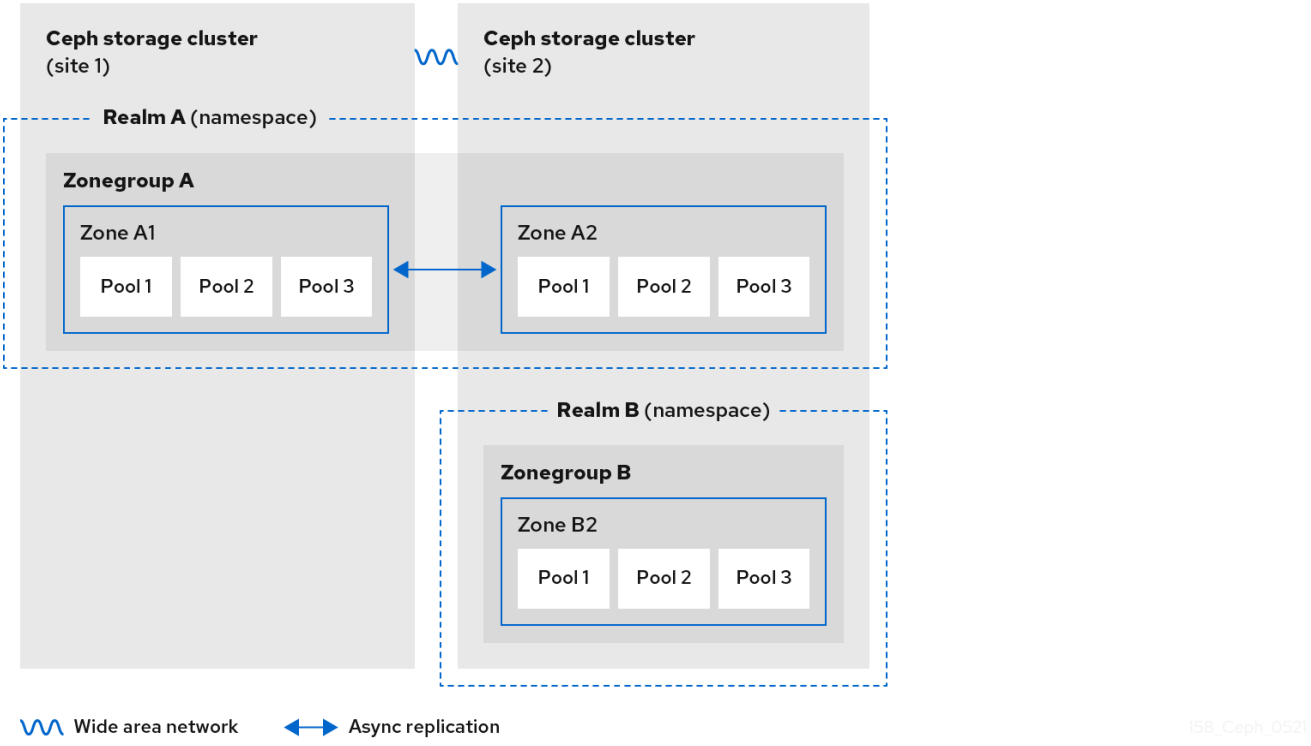


図2.6 2 つのレルムのバリアント



2.7. ストレージのサイズ設定の検討

クラスター設計における最も重要な要因の 1 つは、ストレージ要件 (サイズ調整) を決定することです。Ceph Storage は、ペタバイト以上に拡張できるように設計されています。Ceph Storage クラスターの一般的なサイズの例を以下に示します。

- **小規模:** 250 テラバイト
- **中規模:** 1 ペタバイト
- **大規模:** 2 ペタバイト以上

サイジングには、現在のニーズと近い将来のニーズが含まれます。ゲートウェイクライアントがクラスターに新しいデータを追加する速度を考慮してください。これは、ユースケースごとに異なる可能性があります。たとえば、4k ビデオを録画したり、大量的なイメージを格納したりすると、金額の市場データなど、ストレージ集約型情報よりも速く大量のデータを追加することができます。さらに、レプリケーションやイレイジャーコーディングなどのデータ永続性の方法が、必要なストレージメディアに大きく影響することに注意してください。

サイズ設定の詳細は、Red Hat Ceph Storage ハードウェアガイドおよびそれに関連する OSD ハードウェアの選択に関するリンクを参照してください。

2.8. ストレージの密度の検討

Ceph の設計のもう 1 つの重要な要素には、ストレージの密度があります。通常、ストレージクラスターは、レプリケート、バックフィル、復旧時に適切なパフォーマンスを確保するために、少なくとも 10 個のノードにデータを保存する必要があります。ストレージクラスター内に少なくとも 10 個のノードがあるノードに障害が発生した場合、データの 10% のみが残りのノードに移動する必要があります。ノードの数が大幅に少ない場合は、より高い割合のデータを存続するノードに移動する必要があります。さらに、ストレージクラスターがデータを書き込むことができるように、ノードの障害に対応する

ために **full_ratio** オプションおよび **near_full_ratio** オプションを設定する必要があります。このため、ストレージ密度を考慮することが重要です。ストレージの密度が高いことは必ずしも適切であるとは限りません。

より高いストレージ密度よりも多くのノードを優先するもう 1 つの要因は、イレイジャーコーディングです。イレイジャーコーディングを使用してオブジェクトを作成し、**node** を最小 CRUSH 障害ドメインとして使用する場合、Ceph ストレージクラスターにはデータおよびコーディングチャンクと同じ数のノードが必要になります。たとえば、**k=8, m=3** を使用するクラスターでは、各データまたはコーディングチャンクが別のノードに保存されるように、最低でも 11 個のノードが必要です。

ホットスワップも重要な考慮事項になります。最新のサーバーの多くは、ドライブのホットスワップに対応します。ただし、一部のハードウェア設定では、ドライブを交換するために複数のドライブを取り外す必要があります。Red Hat は、障害が発生したディスクをスワップアウトするときに必要以上の Ceph OSD をダウンさせる可能性があるため、このような設定は回避することを推奨します。

2.9. CEPH MONITOR ノードのディスクの考慮事項

Ceph Monitor は **rocksdb** を使用します。これは同期書き込みのレイテンシーの影響を受けることです。Red Hat は、SSD ディスクを使用して Ceph Monitor データを保存することを強く推奨します。連続した書き込みおよびスループットの特徴が十分にある SSD ディスクを選択します。

2.10. バックフィルとリカバリー設定の調整

I/O は、バックフィルと復旧操作の両方による悪影響を受け、パフォーマンスの低下およびエンドユーザーの不満に繋がります。クラスターの拡張または復旧中の I/O 要求に対応しやすくするには、以下のオプションおよび値を Ceph 設定ファイルに設定します。

```
[osd]
osd_max_backfills = 1
osd_recovery_max_active = 1
osd_recovery_op_priority = 1
```

2.11. クラスターマップサイズの調整

デフォルトでは、**ceph-osd** デーモンは以前の osdmaps を 500 個キャッシュします。重複排除を使用しても、マップはデーモンごとに大量のメモリーを消費する可能性があります。Ceph 設定でキャッシュサイズを調整すると、メモリー消費が大幅に削減される可能性があります。以下に例を示します。

```
[ceph: root@host01 /]# ceph config set global osd_map_message_max 10
[ceph: root@host01 /]# ceph config set osd osd_map_cache_size 20
[ceph: root@host01 /]# ceph config set osd osd_map_share_max_epochs 10
[ceph: root@host01 /]# ceph config set osd osd_pg_epoch_persisted_max_stale 10
```

Red Hat Ceph Storage バージョン 3 以降では、**ceph-manager** デーモンが PG クエリーを処理するため、クラスターマップはパフォーマンスに影響しません。

2.12. スクラビングの調整

デフォルトでは、Ceph はライトスクラブを日単位で実行し、ディープスクラブを週単位で実行します。ライトスクラブは、オブジェクトサイズとチェックサムをチェックして、PG が同じオブジェクトデータを保存していることを確認します。時間の経過とともに、オブジェクトのサイズやチェックサムに関係なく、ディスクセクターが悪化する可能性があります。ディープスクラブは、そのレプリカと共にオブジェクトのコンテンツをチェックして、実際のコンテンツが同じであることを確認します。その

ため、**fsck** の方法で、ディープスクラブがデータの整合性を保ちますが、この手順ではクラスターに I/O のペナルティーを課すことになります。わずかなスクラビングは I/O に影響を及ぼす可能性があります。

デフォルト設定では、Ceph OSD が、ピーク動作時間や高負荷の期間などの不適切な時間にスクラブを開始できます。スクラブ操作がエンドユーザーの操作と競合する場合、エンドユーザーは遅延とパフォーマンスの低下を経験する可能性があります。

エンドユーザーのパフォーマンスの低下を防ぐために、Ceph は、スクラブを負荷の低い期間またはオフピーク時間に制限できるいくつかのスクラブ設定を提供します。詳細は、**Red Hat Ceph Storage Configuration Guide** の Scrubbing the OSD セクションを参照してください。

クラスターで日中に高負荷が発生し、深夜に低負荷が発生する場合は、スクラブを夜間に制限することを検討してください。以下に例を示します。

```
[osd]
osd_scrub_begin_hour = 23   #23:01H, or 10:01PM.
osd_scrub_end_hour = 6      #06:01H or 6:01AM.
```

時間制約がスクラブスケジュールを判断する効果的な方法ではない場合は、**osd_scrub_load_threshold** の使用を検討してください。デフォルト値は **0.5** ですが、負荷が低い場合に変更できます。以下に例を示します。

```
[osd]
osd_scrub_load_threshold = 0.25
```

2.13. OBJECTER_INFLIGHT_OPS を増やします。

スケーラビリティーを向上させるために、未送信の I/O 要求の最大許容数を指定する **objecter_inflight_ops** パラメーターの値を編集できます。このパラメーターは、クライアントトラフィックの制御に使用されます。

```
objecter_inflight_ops = 24576
```

2.14. RGW_THREAD_POOL_SIZE を増やします。

スケーラビリティーを向上させるために、スレッドプールのサイズである **rgw_thread_pool_size** パラメーターの値を編集できます。新しい **beast** フロントエンドは、新しい接続を受け入れるためのスレッドプールサイズの制限を受けません。

```
rgw_thread_pool_size = 512
```

2.15. CEPH 実行時の LINUX カーネルのチューニングに関する考察

実稼働環境用の Red Hat Ceph Storage クラスターでは、一般的にオペレーティングシステムのチューニング (特に制限とメモリー割り当て) が有効です。ストレージクラスター内の全ホストに調整が設定されていることを確認します。また、Red Hat サポートでケースを開き、追加でアドバイスを求めることもできます。

ファイル記述子の増加

Ceph Object Gateway は、ファイル記述子が不足すると停止することがあります。Ceph Object Gateway ホストの **/etc/security/limits.conf** ファイルを変更して、Ceph Object Gateway のファイル記述子を増やすことができます。

```
ceph       soft    nofile     unlimited
```

大規模ストレージクラスターの ulimit 値の調整

たとえば、1024 個以上の Ceph OSD を使用する大規模なストレージクラスターで Ceph 管理コマンドを実行する場合は、次の内容で管理コマンドを実行する各ホストに **/etc/security/limits.d/50-ceph.conf** ファイルを作成します。

```
USER_NAME       soft    nproc     unlimited
```

USER_NAME は、Ceph の管理コマンドを実行する root 以外のユーザーのアカウント名に置き換えます。

注記

Red Hat Enterprise Linux では、root ユーザーの **ulimit** 値はすでにデフォルトで **unlimited** に設定されています。

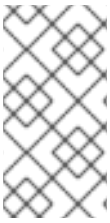
関連情報

- Ceph のさまざまな内部コンポーネントや、それらのコンポーネントに関するストラテジーの詳細は、Red Hat Ceph Storage Storage ストラテジーガイドを参照してください。

第3章 DEPLOYMENT

ストレージ管理者は、Ceph Orchestrator とコマンドラインインターフェイスまたはサービス仕様を使用して Ceph Object Gateway をデプロイすることができます。また、マルチサイト Ceph Object Gateway を設定して、Ceph Orchestrator を使用して Ceph Object Gateway を削除することもできます。

cephadm コマンドは、Ceph Object Gateway を、単一クラスターデプロイメントまたはマルチサイトデプロイメントの特定のレルムおよびゾーンを管理するデーモンのコレクションとしてデプロイします。



注記

cephadm では、Ceph Object Gateway デーモンは、 **ceph.conf** ファイルやコマンドラインオプションではなく、Ceph Monitor 設定データベースを使用して設定されます。設定が **client.rgw** セクションにない場合、Ceph Object Gateway デーモンはデフォルト設定で起動し、ポート **80** にバインドします。

本セクションでは、以下の管理タスクを説明します。

- コマンドラインインターフェイスを使用した Ceph オブジェクトゲートウェイのデプロイ
- サービス仕様を使用した Ceph Object Gateway のデプロイ
- Ceph Orchestrator を使用したマルチサイト Ceph Object Gateway のデプロイ
- Ceph Orchestrator を使用した Ceph Object Gateway の削除
- Ceph Manager rgw モジュールの使用

前提条件

- 実行中、および正常な Red Hat Ceph Storage クラスター
- すべてのノードへの root レベルのアクセス。
- ストレージクラスターで利用できるノード。
- すべてのマネージャー、モニター、および OSD がストレージクラスターにデプロイされます。

3.1. コマンドラインインターフェイスを使用した CEPH オブジェクトゲートウェイのデプロイ

Ceph Orchestrator を使用すると、コマンドラインインターフェイスで **ceph orch** コマンドを使用して Ceph オブジェクトゲートウェイをデプロイできます。

前提条件

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

- すべてのマネージャー、モニター、および OSD デーモンがデプロイされている。

手順

1. Cephadm シェルにログインします。

   例

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

2. Ceph オブジェクトゲートウェイデーモンは、以下の 3 つの方法でデプロイできます。

方法 1

- レルム、ゾーングループ、ゾーンを作成し、ホスト名で配置仕様を使用します。

  a. レルムを作成します。

     構文

     ```
     radosgw-admin realm create --rgw-realm=REALM_NAME --default
     ```

     例

     ```
     [ceph: root@host01 /]# radosgw-admin realm create --rgw-realm=test_realm --default
     ```

  b. ゾーングループを作成します。

     構文

     ```
     radosgw-admin zonegroup create --rgw-zonegroup=ZONE_GROUP_NAME  --master --default
     ```

     例

     ```
     [ceph: root@host01 /]# radosgw-admin zonegroup create --rgw-zonegroup=default  --master --default
     ```

  c. ゾーンを作成します。

     構文

     ```
     radosgw-admin zone create --rgw-zonegroup=ZONE_GROUP_NAME --rgw-zone=ZONE_NAME --master --default
     ```

     例

     ```
     [ceph: root@host01 /]# radosgw-admin zone create --rgw-zonegroup=default --rgw-zone=test_zone --master --default
     ```

  d. 変更をコミットします。

構文

```
radosgw-admin period update --rgw-realm=REALM_NAME --commit
```

例

```
[ceph: root@host01 /]# radosgw-admin period update --rgw-realm=test_realm --commit
```

e. **ceph orch apply** コマンドを実行します。

構文

```
ceph orch apply rgw NAME [--realm=REALM_NAME] [--zone=ZONE_NAME] [--zonegroup=ZONE_GROUP_NAME] --placement="NUMBER_OF_DAEMONS [HOST_NAME_1 HOST_NAME_2]"
```

例

```
[ceph: root@host01 /]# ceph orch apply rgw test --realm=test_realm --zone=test_zone --zonegroup=default --placement="2 host01 host02"
```

方法 2

- 任意のサービス名を使用して、単一のクラスターデプロイメント用に 2 つの Ceph オブジェクトゲートウェイデーモンをデプロイします。

構文

```
ceph orch apply rgw SERVICE_NAME
```

例

```
[ceph: root@host01 /]# ceph orch apply rgw foo
```

方法 3

- ホストのラベル付きセットで任意のサービス名を使用します。

構文

```
ceph orch host label add HOST_NAME_1 LABEL_NAME
ceph orch host label add HOSTNAME_2 LABEL_NAME
ceph orch apply rgw SERVICE_NAME --placement="label:LABEL_NAME count-per-host:NUMBER_OF_DAEMONS" --port=8000
```

注記

NUMBER_OF_DAEMONS は、各ホストにデプロイされる Ceph オブジェクトゲートウェイの数を制御します。追加のコストを発生させずに最高のパフォーマンスを実現するには、この値を 2 に設定します。

例

```
[ceph: root@host01 /]# ceph orch host label add host01 rgw  # the 'rgw' label can be anything
[ceph: root@host01 /]# ceph orch host label add host02 rgw
[ceph: root@host01 /]# ceph orch apply rgw foo --placement="label:rgw count-per-host:2" --
port=8000
```

検証

- サービスをリスト表示します。

  例

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

- ホスト、デーモン、およびプロセスをリスト表示します。

  構文

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

  例

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

3.2. サービス仕様を使用した CEPH OBJECT GATEWAY のデプロイ

Ceph Object Gateway は、デフォルトまたはカスタムのレルム、ゾーン、およびゾーングループいずれかのサービス仕様を使用してデプロイできます。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- ブートストラップされたホストへの root レベルのアクセス。
- ホストがクラスターに追加されている。
- すべてのマネージャー、モニター、および OSD デーモンがデプロイされます。

手順

1. root ユーザーとして、仕様ファイルを作成します。

   例

   ```
   [root@host01 ~]# touch radosgw.yml
   ```

2. **radosgw.yml** ファイルを編集し、デフォルトレルム、ゾーン、およびゾーングループの以下の詳細を追加します。

   構文

```
service_type: rgw
service_id: REALM_NAME.ZONE_NAME
placement:
  hosts:
  - HOST_NAME_1
  - HOST_NAME_2
  count_per_host: NUMBER_OF_DAEMONS
spec:
  rgw_realm: REALM_NAME
  rgw_zone: ZONE_NAME
  rgw_zonegroup: ZONE_GROUP_NAME
  rgw_frontend_port: FRONT_END_PORT
networks:
  -  NETWORK_CIDR # Ceph Object Gateway service binds to a specific network
```

注記

NUMBER_OF_DAEMONS は、各ホストにデプロイされる Ceph Object Gateway の数を制御します。追加のコストを発生させずに最高のパフォーマンスを実現するには、この値を 2 に設定します。

例

```
service_type: rgw
service_id: default
placement:
  hosts:
  - host01
  - host02
  - host03
  count_per_host: 2
spec:
  rgw_realm: default
  rgw_zone: default
  rgw_zonegroup: default
  rgw_frontend_port: 1234
networks:
  - 192.169.142.0/24
```

3. オプション: カスタムレルム、ゾーン、およびゾーングループの場合は、リソースを作成し、**radosgw.yml** ファイルを作成します。

   a. カスタムレルム、ゾーン、およびゾーングループを作成します。

   例

   ```
   [root@host01 ~]# radosgw-admin realm create --rgw-realm=test_realm --default
   [root@host01 ~]# radosgw-admin zonegroup create --rgw-zonegroup=test_zonegroup --default
   [root@host01 ~]# radosgw-admin zone create --rgw-zonegroup=test_zonegroup --rgw-zone=test_zone --default
   [root@host01 ~]# radosgw-admin period update --rgw-realm=test_realm --commit
   ```

   b. 以下の内容で **radosgw.yml** ファイルを作成します。

例

```
service_type: rgw
service_id: test_realm.test_zone
placement:
  hosts:
  - host01
  - host02
  - host03
  count_per_host: 2
spec:
  rgw_realm: test_realm
  rgw_zone: test_zone
  rgw_zonegroup: test_zonegroup
  rgw_frontend_port: 1234
networks:
  - 192.169.142.0/24
```

4. **radosgw.yml** ファイルをコンテナーのディレクトリーにマウントします。

   例

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

   注記

   シェルを終了するたびに、デーモンをデプロイする前にファイルをコンテナーにマウントする必要があります。

5. サービス仕様を使用して Ceph Object Gateway をデプロイします。

   構文

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

   例

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

検証

- サービスをリスト表示します。

  例

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

- ホスト、デーモン、およびプロセスをリスト表示します。

  構文

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

-

例

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

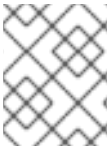
3.3. CEPH ORCHESTRATOR を使用したマルチサイト CEPH OBJECT GATEWAY のデプロイ

Ceph Orchestrator は、Ceph Object Gateway のマルチサイト設定オプションをサポートしています。

各オブジェクトゲートウェイを active-active ゾーン設定で機能するように設定すると、非プライマリーゾーンへの書き込みが可能になります。マルチサイト設定は、レルムと呼ばれるコンテナー内に保存されます。

レルムは、ゾーングループ、ゾーン、および期間を保存します。**rgw** デーモンは同期を処理し、個別の同期エージェントが不要になるため、アクティブ-アクティブ設定で動作します。

コマンドラインインターフェイス (CLI) を使用してマルチサイトゾーンをデプロイすることもできます。



注記

以下の設定では、少なくとも 2 つの Red Hat Ceph Storage クラスターが地理的に別々の場所であることを前提としています。ただし、この設定は同じサイトでも機能します。

前提条件

- 少なくとも 2 つの実行中の Red Hat Ceph Storage クラスター
- 少なくとも 2 つの Ceph Object Gateway インスタンス (各 Red Hat Ceph Storage クラスターに 1 つ)。
- すべてのノードへの root レベルのアクセス。
- ノードまたはコンテナーがストレージクラスターに追加されます。
- すべての Ceph Manager、Monitor、および OSD デーモンがデプロイされます。

手順

1. **cephadm** シェルで、プライマリーゾーンを設定します。
   a. レルムを作成します。

   構文

   ```
   radosgw-admin realm create --rgw-realm=REALM_NAME --default
   ```

   例

   ```
   [ceph: root@host01 /]# radosgw-admin realm create --rgw-realm=test_realm --default
   ```

   ストレージクラスターに単一のレルムがある場合は、**--default** フラグを指定します。

b. プライマリーゾーングループを作成します。

構文

```
radosgw-admin zonegroup create --rgw-zonegroup=ZONE_GROUP_NAME --endpoints=http://RGW_PRIMARY_HOSTNAME:RGW_PRIMARY_PORT_NUMBER_1 --master --default
```

例

```
[ceph: root@host01 /]# radosgw-admin zonegroup create --rgw-zonegroup=us --endpoints=http://rgw1:80 --master --default
```

c. プライマリーゾーンを作成します。

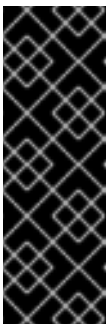
構文

```
radosgw-admin zone create --rgw-zonegroup=PRIMARY_ZONE_GROUP_NAME --rgw-zone=PRIMARY_ZONE_NAME --endpoints=http://RGW_PRIMARY_HOSTNAME:RGW_PRIMARY_PORT_NUMBER_1 --access-key=SYSTEM_ACCESS_KEY --secret=SYSTEM_SECRET_KEY
```

例

```
[ceph: root@host01 /]# radosgw-admin zone create --rgw-zonegroup=us --rgw-zone=us-east-1 --endpoints=http://rgw1:80 --access-key=LIPEYZJLTWXRKXS9LPJC --secret-key=IsAje0AVDNXNw48LjMAimpCpI7VaxJYSnfD0FFKQ
```

d. 必要に応じて、デフォルトゾーン、ゾーングループ、および関連するプールを削除します。



重要

デフォルトゾーンおよびゾーングループを使用してデータを保存する場合は、デフォルトゾーンとそのプールは削除しないでください。また、デフォルトのゾーングループを削除して、システムユーザーを削除します。

default ゾーンおよびゾーングループの古いデータにアクセスするには、**radosgw-admin** コマンドで **--rgw-zone default** および **--rgw-zonegroup default** を使用します。

例

```
[ceph: root@host01 /]# radosgw-admin zonegroup delete --rgw-zonegroup=default
[ceph: root@host01 /]# ceph osd pool rm default.rgw.log default.rgw.log --yes-i-really-really-mean-it
[ceph: root@host01 /]# ceph osd pool rm default.rgw.meta default.rgw.meta --yes-i-really-really-mean-it
[ceph: root@host01 /]# ceph osd pool rm default.rgw.control default.rgw.control --yes-i-really-really-mean-it
[ceph: root@host01 /]# ceph osd pool rm default.rgw.data.root default.rgw.data.root --
```

```
yes-i-really-really-mean-it
[ceph: root@host01 /]# ceph osd pool rm default.rgw.gc default.rgw.gc --yes-i-really-really-mean-it
```

e. システムユーザーを作成します。

構文

```
radosgw-admin user create --uid=USER_NAME --display-name="USER_NAME" --access-key=SYSTEM_ACCESS_KEY --secret=SYSTEM_SECRET_KEY --system
```

例

```
[ceph: root@host01 /]# radosgw-admin user create --uid=zone.user --display-name="Zone user" --system
```

access_key および **secret_key** を書き留めておきます。

f. アクセスキーとシステムキーをプライマリーゾーンに追加します。

構文

```
radosgw-admin zone modify --rgw-zone=PRIMARY_ZONE_NAME --access-key=ACCESS_KEY --secret=SECRET_KEY
```

例

```
[ceph: root@host01 /]# radosgw-admin zone modify --rgw-zone=us-east-1 --access-key=NE48APYCAODEPLKBCZVQ--secret=u24GHQWRE3yxxNBnFBzjM4jn14mFIckQ4EKL6LoW
```

g. 変更をコミットします。

構文

```
radosgw-admin period update --commit
```

例

```
[ceph: root@host01 /]# radosgw-admin period update --commit
```

h. **cephadm** シェルの外部で、ストレージクラスターおよびプロセスの **FSID** を取得します。

例

```
[root@host01 ~]#  systemctl list-units | grep ceph
```

i. Ceph Object Gateway デーモンを起動します。

構文

```
systemctl start ceph-FSID@DAEMON_NAME
systemctl enable ceph-FSID@DAEMON_NAME
```

例

```
[root@host01 ~]# systemctl start ceph-62a081a6-88aa-11eb-a367-
001a4a000672@rgw.test_realm.us-east-1.host01.ahdtsw.service
[root@host01 ~]# systemctl enable ceph-62a081a6-88aa-11eb-a367-
001a4a000672@rgw.test_realm.us-east-1.host01.ahdtsw.service
```

2. Cephadm シェルで、セカンダリーゾーンを設定します。

   a. ホストからプライマリーレルム設定をプルします。

   構文

   ```
   radosgw-admin realm pull --rgw-realm=PRIMARY_REALM --
   url=URL_TO_PRIMARY_ZONE_GATEWAY --access-key=ACCESS_KEY --secret-
   key=SECRET_KEY --default
   ```

   例

   ```
   [ceph: root@host04 /]# radosgw-admin realm pull --rgw-realm=test_realm --
   url=http://10.74.249.26:80 --access-key=LIPEYZJLTWXRKXS9LPJC --secret-
   key=IsAje0AVDNXNw48LjMAimpCpI7VaxJYSnfD0FFKQ --default
   ```

   b. ホストからプライマリー期間設定をプルします。

   構文

   ```
   radosgw-admin period pull --url=URL_TO_PRIMARY_ZONE_GATEWAY --access-
   key=ACCESS_KEY --secret-key=SECRET_KEY
   ```

   例

   ```
   [ceph: root@host04 /]# radosgw-admin period pull --url=http://10.74.249.26:80 --access-
   key=LIPEYZJLTWXRKXS9LPJC --secret-
   key=IsAje0AVDNXNw48LjMAimpCpI7VaxJYSnfD0FFKQ
   ```

   c. セカンダリーゾーンを設定します。

   構文

   ```
   radosgw-admin zone create --rgw-zonegroup=ZONE_GROUP_NAME \
            --rgw-zone=SECONDARY_ZONE_NAME --
   endpoints=http://RGW_SECONDARY_HOSTNAME:RGW_PRIMARY_PORT_NUMBER
   _1 \
            --access-key=SYSTEM_ACCESS_KEY --secret=SYSTEM_SECRET_KEY \
            --endpoints=http://FQDN:80 \
            [--read-only]
   ```

   例

```
[ceph: root@host04 /]# radosgw-admin zone create --rgw-zonegroup=us --rgw-zone=us-east-2 --endpoints=http://rgw2:80 --access-key=LIPEYZJLTWXRKXS9LPJC --secret-key=IsAje0AVDNXNw48LjMAimpCpI7VaxJYSnfD0FFKQ
```

d. 必要に応じて、デフォルトゾーンを削除します。

重要

デフォルトゾーンおよびゾーングループを使用してデータを保存する場合は、デフォルトゾーンとそのプールは削除しないでください。

default ゾーンおよびゾーングループの古いデータにアクセスするには、**radosgw-admin** コマンドで **--rgw-zone default** および **--rgw-zonegroup default** を使用します。

例

```
[ceph: root@host04 /]# radosgw-admin zone rm --rgw-zone=default
[ceph: root@host04 /]# ceph osd pool rm default.rgw.log default.rgw.log --yes-i-really-really-mean-it
[ceph: root@host04 /]# ceph osd pool rm default.rgw.meta default.rgw.meta --yes-i-really-really-mean-it
[ceph: root@host04 /]# ceph osd pool rm default.rgw.control default.rgw.control --yes-i-really-really-mean-it
[ceph: root@host04 /]# ceph osd pool rm default.rgw.data.root default.rgw.data.root --yes-i-really-really-mean-it
[ceph: root@host04 /]# ceph osd pool rm default.rgw.gc default.rgw.gc --yes-i-really-really-mean-it
```

e. Ceph 設定データベースを更新します。

構文

```
ceph config set SERVICE_NAME rgw_zone SECONDARY_ZONE_NAME
```

例

```
[ceph: root@host04 /]# ceph config set rgw rgw_zone us-east-2
```

f. 変更をコミットします。

構文

```
radosgw-admin period update --commit
```

例

```
[ceph: root@host04 /]# radosgw-admin period update --commit
```

g. Cephadm シェルの外部で、ストレージクラスターおよびプロセスの FSID を取得します。

例

```
[root@host04 ~]#  systemctl list-units | grep ceph
```

h. Ceph Object Gateway デーモンを起動します。

構文

```
systemctl start ceph-FSID@DAEMON_NAME
systemctl enable ceph-FSID@DAEMON_NAME
```

例

```
[root@host04 ~]# systemctl start ceph-62a081a6-88aa-11eb-a367-
001a4a000672@rgw.test_realm.us-east-2.host04.ahdtsw.service
[root@host04 ~]# systemctl enable ceph-62a081a6-88aa-11eb-a367-
001a4a000672@rgw.test_realm.us-east-2.host04.ahdtsw.service
```

3. 必要に応じて、配置仕様を使用して、マルチサイトの Ceph Object Gateway をデプロイします。

構文

```
ceph orch apply rgw NAME --realm=REALM_NAME --zone=PRIMARY_ZONE_NAME --
placement="NUMBER_OF_DAEMONS HOST_NAME_1 HOST_NAME_2"
```

例

```
[ceph: root@host04 /]# ceph orch apply rgw east --realm=test_realm --zone=us-east-1 --
placement="2 host01 host02"
```

検証

- 同期のステータスを確認してデプロイメントを確認します。

例

```
[ceph: root@host04 /]# radosgw-admin sync status
```

3.4. CEPH ORCHESTRATOR を使用した CEPH OBJECT GATEWAY の削除

ceph orch rm コマンドを使用して、Ceph オブジェクトゲートウェイデーモンを削除できます。

前提条件

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

- ホストにデプロイされた Ceph オブジェクトゲートウェイデーモンが少なくとも 1 つ含まれます。

手順

1. Cephadm シェルにログインします。

   例

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

2. サービスをリスト表示します。

   例

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

3. サービスを削除します。

   構文

   ```
   ceph orch rm SERVICE_NAME
   ```

   例

   ```
   [ceph: root@host01 /]# ceph orch rm rgw.test_realm.test_zone_bb
   ```

検証

- ホスト、デーモン、およびプロセスをリスト表示します。

  構文

  ```
  ceph orch ps
  ```

  例

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

関連情報

- 詳細は、**Red Hat Ceph Storage オペレーションガイド** の **コマンドラインインターフェイスを使用した Ceph オブジェクトゲートウェイのデプロイメント** セクションを参照してください。
- 詳細は、**Red Hat Ceph Storageオペレーションガイド**の **サービス仕様を使用して Ceph オブジェクトゲートウェイのデプロイ** セクションを参照してください。

3.5. CEPH MANAGER RGW モジュールの使用

ストレージ管理者は、**rgw** モジュールを使用して、単一サイトおよびマルチサイトに Ceph Object Gateway をデプロイできます。これは、Ceph オブジェクトレルム、ゾーングループ、およびさまざまな関連エンティティーのブートストラップと設定に役立ちます。

新しく作成されたレルムまたは既存のレルムに使用可能なトークンを使用できます。このトークンは、レルム情報とそのマスターゾーンエンドポイント認証データをカプセル化する Base64 文字列です。

マルチサイト設定では、これらのトークンを使用してレルムをプルし、**rgwzone create** コマンドを使用してプライマリークラスターのマスターゾーンと同期するセカンダリーゾーンを別のクラスターに作成できます。

3.5.1. rgw モジュールを使用した Ceph Object Gateway のデプロイ

Ceph Object Gateway レルムをブートストラップすると、新しいレルムエンティティー、新しいゾーングループ、および新しいゾーンが作成されます。**rgw** モジュールは、対応する Ceph Object Gateway デーモンを作成してデプロイするようにオーケストレーターに指示します。

ceph mgr module enable rgw コマンドを使用して、**rgw** モジュールを有効にします。**rgw** モジュールを有効にした後、コマンドラインで引数を渡すか、**yaml** 仕様ファイルを使用してレルムをブートストラップします。

前提条件

- 少なくとも 1 つの OSD がデプロイされた実行中の Red Hat Ceph Storage クラスター。

手順

1. Cephadm シェルにログインします。

   例

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

2. rgw モジュールを有効にします。

   例

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

3. コマンドラインまたは yaml 仕様ファイルを使用して、Ceph Object Gateway レルムをブートストラップします。

   - オプション 1: コマンドラインインターフェイスを使用します。

     構文

     ```
     ceph rgw realm bootstrap [--realm name REALM_NAME] [--zonegroup-name ZONEGROUP_NAME] [--zone-name ZONE_NAME] [--port PORT_NUMBER] [--placement HOSTNAME] [--start-radosgw]
     ```

     例

     ```
     [ceph: root@host01 /]# ceph rgw realm bootstrap --realm-name myrealm --zonegroup-name myzonegroup --zone-name myzone --port 5500 --placement="host01 host02" --
     ```

```
start-radosgw
Realm(s) created correctly. Please, use 'ceph rgw realm tokens' to get the token.
```

- オプション 2: yaml 仕様ファイルを使用します。

  a. root ユーザーとして、yaml ファイルを作成します。

  構文

  ```
  rgw_realm: REALM_NAME
  rgw_zonegroup: ZONEGROUP_NAME
  rgw_zone: ZONE_NAME
  placement:
    hosts:
     - HOSTNAME_1
     - HOSTNAME_2
  ```

  例

  ```
  [root@host01 ~]# cat rgw.yaml

  rgw_realm: myrealm
  rgw_zonegroup: myzonegroup
  rgw_zone: myzone
  placement:
    hosts:
     - host01
     - host02
  ```

  b. YAML ファイルをコンテナー内のディレクトリーにマウントします。

  例

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

  c. レルムをブートストラップします。

  例

  ```
  [ceph: root@host01 /]# ceph rgw realm bootstrap -i /var/lib/ceph/rgw/rgw.yaml
  ```

  注記

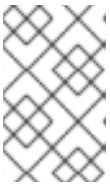
  rgw モジュールで使用される仕様ファイルの形式は、オーケストレーターで使用されるものと同じです。したがって、SSL 証明書などの高度な設定機能を含む、オーケストレーションがサポートする Ceph Object Gateway パラメーターを提供できます。

4. 利用可能なトークンをリストします。

例

```
[ceph: root@host01 /]# ceph rgw realm tokens | jq
```

```
[
  {
    "realm": "myrealm",
    "token":
"ewogICAgInJlYWxtX25hbWUiOiAibXlyZWFsbSIsCiAgICAicmVhbG1faWQiOiAiZDA3YzAwZW
YtOTA0MS00ZjZlLTg4MDQtN2Q0MDI0MDU1NmFlIiwKICAgICJlbmRwb2ludCI6ICJodHRwOi
8vdm0tMDA6NDMyMSIsCiAgICAiYWNjZXNzX2tleSI6ICI5NTY1VFZSMVFWTExFRzdVNFIxR
CIsCiAgICAic2VjcmV0IjogImQ3b0FJQXZrNEdYeXpyd3Q2QVZ6bEZNQmNnRG53RVdMMHF
DenE3cjUiCn1="
  }
]
```



注記

Ceph Object Gateway デーモンがデプロイされる前に上記のコマンドを実行すると、まだエンドポイントがないためトークンがないという旨のメッセージが表示されます。

検証

- オブジェクトゲートウェイのデプロイメントを確認します。

  例

```
[ceph: root@host01 /]# ceph orch list --daemon-type=rgw
NAME                                                HOST                                PORTS  STATUS
REFRESHED   AGE  MEM USE  MEM LIM  VERSION          IMAGE ID      CONTAINER ID
rgw.myrealm.myzonegroup.ceph-saya-6-osd-host01.eburst  ceph-saya-6-osd-host01  *:80
running (111m)     9m ago  111m     82.3M        -  17.2.6-22.el9cp  2d5b080de0b0
2f3eaca7e88e
```

3.5.2. rgw モジュールを使用した Ceph Object Gateway マルチサイトのデプロイ

Ceph Object Gateway レルムをブートストラップすると、新しいレルムエンティティー、新しいゾーングループ、および新しいゾーンが作成されます。マルチサイト同期操作に使用できる新しいシステムユーザーを設定します。**rgw** モジュールは、対応する Ceph Object Gateway デーモンを作成してデプロイするようにオーケストレーターに指示します。

ceph mgr module enable rgw コマンドを使用して、**rgw** モジュールを有効にします。**rgw** モジュールを有効にした後、コマンドラインで引数を渡すか、**yaml** 仕様ファイルを使用してレルムをブートストラップします。

前提条件

- 少なくとも 1 つの OSD がデプロイされた実行中の Red Hat Ceph Storage クラスター。

手順

1. Cephadm シェルにログインします。

   例

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

2. rgw モジュールを有効にします。

例

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

3. コマンドラインまたは yaml 仕様ファイルを使用して、Ceph Object Gateway レルムをブートストラップします。

- オプション 1: コマンドラインインターフェイスを使用します。

構文

```
ceph rgw realm bootstrap [--realm name REALM_NAME] [--zonegroup-name ZONEGROUP_NAME] [--zone-name ZONE_NAME] [--port PORT_NUMBER] [--placement HOSTNAME] [--start-radosgw]
```

例

```
[ceph: root@host01 /]# ceph rgw realm bootstrap --realm-name myrealm --zonegroup-name myzonegroup --zone-name myzone --port 5500 --placement="host01 host02" --start-radosgw
Realm(s) created correctly. Please, use 'ceph rgw realm tokens' to get the token.
```

- オプション 2: yaml 仕様ファイルを使用します。

a. root ユーザーとして、yaml ファイルを作成します。

構文

```
rgw_realm: REALM_NAME
rgw_zonegroup: ZONEGROUP_NAME
rgw_zone: ZONE_NAME
placement:
  hosts:
   - HOSTNAME_1
   - HOSTNAME_2
spec:
  rgw_frontend_port: PORT_NUMBER
zone_endpoints: http://RGW_HOSTNAME_1:RGW_PORT_NUMBER_1,
http://RGW_HOSTNAME_2:RGW_PORT_NUMBER_2
```

例

```
[root@host01 ~]# cat rgw.yaml

rgw_realm: myrealm
rgw_zonegroup: myzonegroup
rgw_zone: myzone
placement:
  hosts:
```

```
   - host01
   - host02
spec:
  rgw_frontend_port: 5500
zone_endpoints: http://<rgw_host1>:<rgw_port1>, http://<rgw_host2>:<rgw_port2>
```

b. YAML ファイルをコンテナー内のディレクトリーにマウントします。

例

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

c. レルムをブートストラップします。

例

```
[ceph: root@host01 /]# ceph rgw realm bootstrap -i /var/lib/ceph/rgw/rgw.yaml
```

注記

rgw モジュールで使用される仕様ファイルの形式は、オーケストレーターで使用されるものと同じです。したがって、SSL 証明書などの高度な設定機能を含む、オーケストレーションがサポートする Ceph Object Gateway パラメーターを提供できます。

4. 利用可能なトークンをリストします。

例

```
[ceph: root@host01 /]# ceph rgw realm tokens | jq

[
  {
    "realm": "myrealm",
    "token":
"ewogICAgInJlYWxtX25hbWUiOiAibXlyZWFsbSIsCiAgICAicmVhbG1faWQiOiAiZDA3YzAwZW
YtOTA0MS00ZjZlLTg4MDQtN2Q0MDI0MDU1NmFlIiwKICAgICJlbmRwb2ludCI6ICJodHRwOi
8vdm0tMDA6NDMyMSIsCiAgICAiYWNjZXNzX2tleSI6ICI5NTY1VFZSMVFWTExFRzdVNFIxR
CIsCiAgICAic2VjcmV0IjogImQ3b0FJQXZrNEdYeXpyd3Q2QVZ6bEZNQmNnRG53RVdMMHF
DenE3cjUiCn1="
  }
]
```

注記

Ceph Object Gateway デーモンがデプロイされる前に上記のコマンドを実行すると、まだエンドポイントがないためトークンがないという旨のメッセージが表示されます。

5. これらのトークンを使用してセカンダリーゾーンを作成し、既存のレルムに参加します。

a. root ユーザーとして、yaml ファイルを作成します。

例

```
[root@host01 ~]# cat zone-spec.yaml
rgw_zone: my-secondary-zone
rgw_realm_token: <token>
placement:
  hosts:
   - ceph-node-1
   - ceph-node-2
spec:
  rgw_frontend_port: 5500
```

b. コンテナー内のディレクトリーに **zone-spec.yaml** ファイルをマウントします。

例

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

c. セカンダリーゾーンで rgw モジュールを有効にします。

例

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

d. セカンダリーゾーンを作成します。

例

```
[ceph: root@host01 /]# ceph rgw zone create -i /var/lib/ceph/radosgw/zone-spec.yaml
```

検証

- Object Gateway のマルチサイトデプロイメントを確認します。

例

```
[ceph: root@host01 /]# radosgw-admin realm list
{
"default_info": "d07c00ef-9041-4f6e-8804-7d40240556ae",
"realms": [
"myrealm"
]
}
```

第4章 BASIC CONFIGURATION

ストレージ管理者として、Ceph Object Gateway の設定の基本を理解することが重要です。Beast と呼ばれるデフォルトの組み込み Web サーバーについて学ぶことができます。Ceph Object Gateway の問題のトラブルシューティングについては、Ceph Object Gateway によって生成されるロギングおよびデバッグ出力を調整できます。また、Ceph Object Gateway を使用してストレージクラスターアクセスに高可用性プロキシーも提供できます。

前提条件

- 実行中、および正常な Red Hat Ceph Storage クラスター
- Ceph Object Gateway ソフトウェアパッケージのインストール

4.1. DNS へのワイルドカードの追加

ホスト名などのワイルドカードを DNS サーバーの DNS レコードに追加できます。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- Ceph Object Gateway がインストールされている。
- 管理ノードへのルートレベルのアクセス。

手順

1. S3 スタイルのサブドメインで Ceph を使用するには、**ceph-radosgw** デーモンがドメイン名を解決するために使用する DNS サーバーの DNS レコードにワイルドカードを追加します。

   構文

   ```
   bucket-name.domain-name.com
   ```

   dnsmasq の場合は、ホスト名の先頭にドット (.) を付けた以下のアドレス設定を追加します。

   構文

   ```
   address=/.HOSTNAME_OR_FQDN/HOST_IP_ADDRESS
   ```

   例

   ```
   address=/.gateway-host01/192.168.122.75
   ```

   bind の場合は、ワイルドカードを DNS レコードに追加します。

   例

   ```
   $TTL    604800
   @       IN      SOA     gateway-host01. root.gateway-host01. (
                                 2         ; Serial
                            604800         ; Refresh
   ```

```
                    86400         ; Retry
                   2419200         ; Expire
                    604800 )       ; Negative Cache TTL
;
@       IN      NS      gateway-host01.
@       IN      A       192.168.122.113
*       IN      CNAME   @
```

2. DNS サーバーを再起動して、サブドメインを使用してサーバーに ping し、**ceph-radosgw** デーモンがサブドメイン要求を処理できるようにします。

   構文

   ```
   ping mybucket.HOSTNAME
   ```

   例

   ```
   [root@host01 ~]# ping mybucket.gateway-host01
   ```

3. DNS サーバーがローカルマシンにある場合は、ローカルマシンのネームサーバーエントリーを追加して **/etc/resolv.conf** を変更しないといけない場合があります。

4. Ceph Object Gateway ゾーングループにホスト名を追加します。

   a. ゾーングループを取得します。

      構文

      ```
      radosgw-admin zonegroup get --rgw-zonegroup=ZONEGROUP_NAME > zonegroup.json
      ```

      例

      ```
      [ceph: root@host01 /]# radosgw-admin zonegroup get --rgw-zonegroup=us > zonegroup.json
      ```

   b. JSON ファイルのバックアップを取ります。

      例

      ```
      [ceph: root@host01 /]# cp zonegroup.json zonegroup.backup.json
      ```

   c. **zonegroup.json** ファイルを表示します。

      例

      ```
      [ceph: root@host01 /]# cat zonegroup.json
      {
          "id": "d523b624-2fa5-4412-92d5-a739245f0451",
          "name": "asia",
          "api_name": "asia",
          "is_master": "true",
          "endpoints": [],
      ```

```
    "hostnames": [],
    "hostnames_s3website": [],
    "master_zone": "d2a3b90f-f4f3-4d38-ac1f-6463a2b93c32",
    "zones": [
        {
            "id": "d2a3b90f-f4f3-4d38-ac1f-6463a2b93c32",
            "name": "india",
            "endpoints": [],
            "log_meta": "false",
            "log_data": "false",
            "bucket_index_max_shards": 11,
            "read_only": "false",
            "tier_type": "",
            "sync_from_all": "true",
            "sync_from": [],
            "redirect_zone": ""
        }
    ],
    "placement_targets": [
        {
            "name": "default-placement",
            "tags": [],
            "storage_classes": [
                "STANDARD"
            ]
        }
    ],
    "default_placement": "default-placement",
    "realm_id": "d7e2ad25-1630-4aee-9627-84f24e13017f",
    "sync_policy": {
        "groups": []
    }
}
```

d. **zonegroup.json** ファイルを新しいホスト名で更新します。

例

```
"hostnames": ["host01", "host02","host03"],
```

e. ゾーングループを Ceph Object Gateway に戻します。

構文

```
radosgw-admin zonegroup set --rgw-zonegroup=ZONEGROUP_NAME --
infile=zonegroup.json
```

例

```
[ceph: root@host01 /]# radosgw-admin zonegroup set --rgw-zonegroup=us --
infile=zonegroup.json
```

f. 期間を更新します。

例

```
[ceph: root@host01 /]# radosgw-admin period update --commit
```

g. Ceph Object Gateway を再起動して DNS 設定を有効にします。

関連情報

- 詳細は、**Red Hat Ceph Storage 設定ガイド** の **Ceph 設定データベース** セクションを参照してください。

4.2. BEAST フロントエンド WEB サーバー

Ceph Object Gateway は、C/C 埋め込みフロントエンド Web サーバーである Beast を提供します。Beast は `Boost.Beast` C ライブラリーを使用して HTTP を解析し、**Boost.Asio** を使用して非同期ネットワーク I/O を行います。

関連情報

- Boost C++ ライブラリー

4.3. BEAST 設定オプション

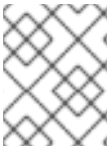
以下の Beast 設定オプションは、RADOS Gateway の Ceph 設定ファイルの組み込み Web サーバーに渡すことができます。それぞれのオプションにはデフォルト値があります。値の指定がない場合は、デフォルト値が空になります。

オプション	説明	デフォルト
endpoint および **ssl_endpoint**	**address[:port]** 形式のリスニングアドレスを設定します。アドレスはドット付き 10 進数形式の IPv4 アドレス文字列、または 16 進数表記の IPv6 アドレスが角括弧で囲まれている IPv6 アドレスです。任意のポートのデフォルトは、**endpoint** が **8080** になり、**ssl_endpoint** が **443** になります。**endpoint=[::1] endpoint=192.168.0.100:8000** のように複数回指定できます。	空
ssl_certificate	SSL 対応のエンドポイントに使用する SSL 証明書ファイルへのパス。	空
ssl_private_key	SSL 対応のエンドポイントに使用される秘密鍵ファイルへのオプションのパス。**ssl_certificate** で指定されているファイルがない場合は、秘密鍵として使用されます。	空
tcp_nodelay	一部の環境でのパフォーマンスの最適化。	空

SSL を使用する Beast オプションのある /etc/ceph/ceph.conf ファイルの例:

```
...
```

```
[client.rgw.node1]
rgw frontends = beast ssl_endpoint=192.168.0.100:443 ssl_certificate=<path to SSL certificate>
```



注記

デフォルトでは、Beast フロントエンドは、サーバーによって処理されるすべての要求を記録するアクセスログラインを RADOS Gateway ログファイルに書き込みます。

関連情報

- 詳細は、Beast フロントエンド を参照してください。

4.4. BEAST の SSL の設定

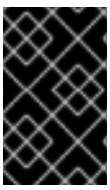
Beast フロントエンド Web サーバーが OpenSSL ライブラリーを使用して Transport Layer Security (TLS) を提供するように設定できます。Beast で Secure Socket Layer (SSL) を使用するには、Ceph Object Gateway ノードのホスト名と一致する認証局 (CA) から証明書を取得する必要があります。また、Beast は、1 つ **.pem** ファイルに秘密鍵、サーバー証明書、およびその他の CA を含める必要があります。

重要

秘密鍵ハッシュが含まれているため、**.pem** ファイルへ不正アクセスされないようにします。

重要

Red Hat は、SAN (Subject Alternative Name) フィールドと S3 スタイルのサブドメインで使用するワイルドカードを使用して CA から証明書を取得することを推奨します。



重要

Red Hat は、小規模および中規模のテスト環境で、Beast フロントエンド Web サーバーで SSL のみを使用することを推奨します。実稼働環境では、HAProxy で SSL 接続を終了するには HAProxy および **keepalived** を使用する必要があります。

現在、カスタム CA を Ceph Object Gateway に挿入できないため、Ceph Object Gateway がクライアントとして機能し、サーバーでカスタム証明書が使用されている場合は、**rgw_verify_ssl** パラメーターを **false** に設定します。

例

```
[ceph: root@host01 /]# ceph config set client.rgw rgw_verify_ssl false
```

前提条件

- 実行中、および正常な Red Hat Ceph Storage クラスター
- Ceph Object Gateway ソフトウェアパッケージのインストール
- OpenSSL ソフトウェアパッケージのインストール

- Ceph Object Gateway ノードへのルートレベルのアクセスがある。

手順

1. 現在のディレクトリーに **rgw.yml** という名前の新規ファイルを作成します。

   例

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

2. 編集する **rgw.yml** ファイルを開き、環境に合わせてカスタマイズします。

   構文

   ```
   service_type: rgw
   service_id: SERVICE_ID
   service_name: SERVICE_NAME
   placement:
     hosts:
     - HOST_NAME
   spec:
     ssl: true
     rgw_frontend_ssl_certificate: CERT_HASH
   ```

   例

   ```
   service_type: rgw
   service_id: foo
   service_name: rgw.foo
   placement:
     hosts:
     - host01
   spec:
     ssl: true
     rgw_frontend_ssl_certificate: |
       -----BEGIN RSA PRIVATE KEY-----
       MIIEpAIBAAKCAQEA+Cf4l9OagD6x67HhdCy4Asqw89Zz9ZuGbH50/7ltIMQpJJU0
       gu9ObNtIoC0zabJ7n1jujueYgIpOqGnhRSvsGJiEkgN81NLQ9rqAVaGpadjrNLcM
       bpgqJCZj0vzzmtFBCtenpb5l/EccMFcAydGtGeLP33SaWiZ4Rne56GBInk6SATI/
       JSKweGD1y5GiAWipBR4C74HiAW9q6hCOuSdp/2WQxWT3T1j2sjlqxkHdtInUtwOm
       j5Ism276IndeQ9hR3reFR8PJnKIPx73oTBQ7p9CMR1J4ucq9Ny0J12wQYT00fmJp
       -----END RSA PRIVATE KEY-----
       -----BEGIN CERTIFICATE-----
       MIIEBTCCAu2gAwIBAgIUGfYFsj8HyA9Zv2l600hxzT8+gG4wDQYJKoZIhvcNAQEL

   BQAwgYkxCzAJBgNVBAYTAklOMQwwCgYDVQQIDANLQVIxDDAKBgNVBAcMA0JMUjEM

   MAoGA1UECgwDUkhUMQswCQYDVQQLDAJCVTEkMCIGA1UEAwwbY2VwaC1zc2wtcmhj
       czUtOGRjeHY2LW5vZGU1MR0wGwYJKoZIhvcNAQkBFg5hYmNAcmVkaGF0LmNvbTAe
       -----END CERTIFICATE-----
   ```

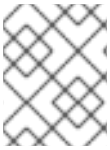
3. サービス仕様ファイルを使用して Ceph Object Gateway をデプロイします。

   例

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

4.5. D3N データキャッシュ

データセンターデータ配信ネットワーク (D3N) は、**NVMe** などの高速ストレージを使用して、アクセス側でデータセットをキャッシュします。このようなキャッシュにより、ビッグデータジョブは、エッジサイトの各 Rados Gateway ノードで利用可能なコンピューティングリソースと高速ストレージリソースを使用できるようになります。Rados ゲートウェイは、バックエンドオブジェクトストア (OSD) のキャッシュサーバーとして機能し、再利用のためにデータをローカルに保存します。



注記

Rados ゲートウェイが再起動されるたびに、キャッシュディレクトリーの内容が消去されます。

4.5.1. D3N キャッシュディレクトリーの追加

RGW で D3N キャッシュを有効にするには、**podmanunit.run** に D3N キャッシュディレクトリーを含める必要もあります。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- Ceph Object Gateway がインストールされている。
- 管理ノードへのルートレベルのアクセス。
- 各 RGW ノード内の高速 NVMe ドライブは、ローカルキャッシュストレージとして機能する。

手順

1. NVMe ドライブのマウントポイントを作成します。

   構文

   ```
   mkfs.ext4 nvme-drive-path
   ```

   例

   ```
   [ceph: root@host01 /]# mkfs.ext4 /dev/nvme0n1
   mount /dev/nvme0n1 /mnt/nvme0n1/
   ```

2. キャッシュディレクトリーのパスを作成します。

   構文

   ```
   mkdir <nvme-mount-path>/cache-directory-name
   ```

   例

   ```
   [ceph: root@host01 /]# mkdir /mnt/nvme0n1/rgw_datacache
   ```

3. **nvme-mount-path** および **rgw_d3n_l1_datacache_persistent_path** に +rwx 権限を付与します。

   構文

   ```
   chmod a+rwx nvme-mount-path ; chmod a+rwx rgw_d3n_l1_datacache_persistent_path
   ```

   例

   ```
   [ceph: root@host01 /]# chmod a+rwx /mnt/nvme0n1 ; chmod a+rwx
   /mnt/nvme0n1/rgw_datacache/
   ```

4. **extra_container_args** を使用して RGW 仕様ファイルを作成または変更し、**rgw_d3n_l1_datacache_persistent_path** を **podman Unit.run** に追加します。

   構文

   ```
   "extra_container_args:
           "-v"
           "rgw_d3n_l1_datacache_persistent_path:rgw_d3n_l1_datacache_persistent_path"
     "
   ```

   例

   ```
   [ceph: root@host01 /]# cat rgw-spec.yml
    service_type: rgw
    service_id: rgw.test
    placement:
     hosts:
        host1
        host2
     extra_container_args:
        "-v"
        "/mnt/nvme0n1/rgw_datacache/:/mnt/nvme0n1/rgw_datacache/"
   ```

注記

単一ホストに RGW の複数のインスタンスがある場合は、インスタンスごとに個別の **rgw_d3n_l1_datacache_persistent_path** を作成し、各パスを **extra_container_args** に追加する必要があります。

例:

各ホストの RGW の 2 つのインスタンスの場合、**rgw_d3n_l1_datacache_persistent_path** の下に 2 つの個別の **キャッシュディレクトリー**、**/mnt/nvme0n1/rgw_datacache/rgw1** および **/mnt/nvme0n1/rgw_datacache/rgw2** を作成します。

rgw 仕様ファイルの extra_container_args の例:

```
"extra_container_args:
 "-v"
 "/mnt/nvme0n1/rgw_datacache/rgw1/:/mnt/nvme0n1/rgw_datacache/rgw1/"
 "-v"
 "/mnt/nvme0n1/rgw_datacache/rgw2/:/mnt/nvme0n1/rgw_datacache/rgw2/"
"
```

rgw-spec.yml の例:

```
[ceph: root@host01 /]# cat rgw-spec.yml
 service_type: rgw
  service_id: rgw.test
  placement:
   hosts:
    host1
    host2
   count_per_host: 2
   extra_container_args:
    "-v"
    "/mnt/nvme0n1/rgw_datacache/rgw1/:/mnt/nvme0n1/rgw_datacache/rgw1/"
    "-v"
    "/mnt/nvme0n1/rgw_datacache/rgw2/:/mnt/nvme0n1/rgw_datacache/rgw2/"
```

5. RGW サービスを再デプロイします。

   例

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

4.5.2. rados ゲートウェイでの D3N の設定

既存の RGW で D3N データキャッシュを設定して、Red Hat Ceph Storage クラスターで実行されるビッグデータジョブのパフォーマンスを向上させることができます。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- Ceph Object Gateway がインストールされている。

- 管理ノードへのルートレベルのアクセス。
- キャッシュストレージとして機能する高速 NVMe。

必要な D3N 関連の設定の追加

既存の RGW で D3N を有効にするには、各 Rados Gateways クライアントに対して次の設定を設定する必要があります。

構文

```
# ceph config set <client.rgw> <CONF-OPTION> <VALUE>
```

- **rgw_d3n_l1_local_datacache_enabled=true**
- **rgw_d3n_l1_datacache_persistent_path=path to the cache directory**

  例

  ```
  rgw_d3n_l1_datacache_persistent_path=/mnt/nvme/rgw_datacache/
  ```

- **rgw_d3n_l1_datacache_size=max_size_of_cache_in_bytes**

  例

  ```
  rgw_d3n_l1_datacache_size=10737418240
  ```

手順の例

1. テストオブジェクトを作成します。

   注記

   テストオブジェクトはキャッシュするには 4 MB より大きい必要があります。

   例

   ```
   [ceph: root@host01 /]# fallocate -l 1G ./1G.dat
   [ceph: root@host01 /]# s3cmd mb s3://bkt
   [ceph: root@host01 /]# s3cmd put ./1G.dat s3://bkt
   ```

2. オブジェクトの **GET** を実行します。

   例

   ```
   [ceph: root@host01 /]# s3cmd get s3://bkt/1G.dat /dev/shm/1G_get.dat
   download: 's3://bkt/1G.dat' -> './1G_get.dat'  [1 of 1]
   1073741824 of 1073741824   100% in    13s    73.94 MB/s  done
   ```

3. キャッシュの作成を確認します。キャッシュは、設定された **rgw_d3n_l1_datacache_persistent_path** 内のオブジェクト **key-name** で設定される名前で作成されます。

   例

例

```
[ceph: root@host01 /]# ls -lh /mnt/nvme/rgw_datacache

rw-rr. 1 ceph ceph 1.0M Jun  2 06:18 cc7f967c-0021-43b2-9fdf-
23858e868663.615391.1_shadow.ZCiCtMWeu_19wb100JIEZ-o4tv2lyA_1
```

4. オブジェクトのキャッシュが作成されると、そのオブジェクトに対する次の **GET** 操作はキャッシュからアクセスされるため、アクセスが高速になります。

例

```
[ceph: root@host01 /]# s3cmd get s3://bkt/1G.dat /dev/shm/1G_get.dat
download: 's3://bkt/1G.dat' -> './1G_get.dat'  [1 of 1]
1073741824 of 1073741824   100% in    6s   155.07 MB/s  done
```

上記の例では、キャッシュの高速化を示すために、RAM ドライブ (**/dev/shm**) に書き込みを行っています。

関連情報

- 高可用性の使用に関する詳細は、**Red Hat Ceph Storage トラブルシューティングガイド** の **Ceph サブシステムのデフォルトのログレベル値** セクションを参照してください。
- 高可用性の使用に関する詳細は、**Red Hat Ceph Storage トラブルシューティングガイド** の **Ceph ログについて** セクションを参照してください。

4.6. ロギングおよびデバッグ出力の調整

設定手順を完了したら、ログの出力を確認して、ニーズを満たしていることを確認してください。デフォルトでは、Ceph デーモンは **journald** にログを記録し、**journalctl** コマンドを使用してログを表示できます。または、Ceph デーモンは **/var/log/ceph/CEPH_CLUSTER_ID/** ディレクトリーにあるファイルにログを記録することもできます。

重要

詳細なロギングは、1時間あたり1GB を超えるデータを生成することができます。このタイプのログは、オペレーティングシステムのディスクを満杯にして、オペレーティングシステムの機能を停止させる可能性があります。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- Ceph Object Gateway ソフトウェアのインストール。

手順

1. Ceph Object Gateway のロギングの出力を増やすには、以下のパラメーターを設定します。

構文

```
ceph config set client.rgw debug_rgw VALUE
```

例

```
[ceph: root@host01 /]# ceph config set client.rgw debug_rgw 20
```

a. 起動時にこれらの設定を変更することもできます。

構文

```
ceph --admin-daemon /var/run/ceph/ceph-client.rgw.NAME.asok config set debug_rgw
VALUE
```

例

```
[ceph: root@host01 /]# ceph --admin-daemon /var/run/ceph/ceph-client.rgw.rgw.asok
config set debug_rgw 20
```

2. 必要に応じて、Ceph デーモンを設定して、出力をファイルに記録することができます。**log_to_file** オプションおよび **mon_cluster_log_to_file** オプションを **true** に設定します。

例

```
[ceph: root@host01 /]# ceph config set global log_to_file true
[ceph: root@host01 /]# ceph config set global mon_cluster_log_to_file true
```

関連情報

- 詳細は、**Red Hat Ceph Storage 設定ガイド**の Ceph デバッグおよびロギング設定 セクションを参照してください。

4.7. 静的 WEB ホスト

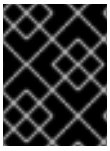
ストレージ管理者は、S3 バケットで Ceph Object Gateway を静的 Web サイトをホストするように設定できます。従来の Web サイトのホストでは、Web サーバーごとに Web サーバーを設定し、コンテンツが動的に変更されない場合にリソースを非効率に使用することができます。たとえば、PHP、servlets、databases、nodejs などのサーバー側のサービスを使用しないサイト。このアプローチは、サイトごとに Web サーバーを備えた仮想マシンをセットアップするよりもはるかに経済的です。

前提条件

- 正常かつ実行中の Red Hat Ceph Storage クラスター

4.7.1. 静的 Web ホストの前提条件

静的 Web ホストには、少なくとも Red Hat Ceph Storage クラスター 1 台と、静的な Web サイト用に少なくとも 2 つの Ceph Object Gateway インスタンスが必要です。Red Hat は、各ゾーンに高可用性 (HA) プロキシーおよび **keepalived** などのロードバランサーを使用する複数のゲートウェイインスタンスがあることを前提としています。



重要

Red Hat は、Ceph Object Gateway インスタンスを使用した標準の S3/Swift API と静的 Web ホストの両方を同時にデプロイすることはサポート **されません**。

関連情報

- 高可用性の使用について、詳しくは **Red Hat Ceph Storage オブジェクトゲートウェイガイド** の **高可用性サービス** セクションを参照してください。

4.7.2. 静的 Web ホストの要件

静的 Web ホスティング機能は独自の API を使用するため、S3 バケットで静的 Web サイトを使用するようにゲートウェイを設定するには、以下が必要です。

1. S3 静的 Web ホストでは、Ceph Object Gateway インスタンスが使用され、標準の S3/Swift API のユースケースに使用されるインスタンスと区別されます。
2. S3 静的 Web サイトをホストするゲートウェイインスタンスは、標準の S3/Swift API ゲートウェイインスタンスとは別の重複しないドメイン名を持っている必要があります。
3. S3 静的 Web サイトをホストするゲートウェイインスタンスは、標準の S3/Swift API ゲートウェイインスタンスとは別のパブリック向け IP アドレスを使用する必要があります。
4. S3 静的 Web サイトをホストするゲートウェイインスタンスは負荷分散し、必要に応じて HAProxy/keepalived を使用して SSL を終了します。

4.7.3. 静的 Web ホストゲートウェイの設定

静的 Web ホスト用に Ceph Object Gateway を有効にするには、以下のオプションを設定します。

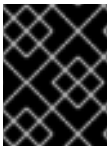
構文

```
ceph config set client.rgw OPTION VALUE
```

例

```
[ceph: root@host01 /]# ceph config set client.rgw rgw_enable_static_website true
[ceph: root@host01 /]# ceph config set client.rgw rgw_enable_apis s3,s3website
[ceph: root@host01 /]# ceph config set client.rgw rgw_dns_name objects-zonegroup.example.com
[ceph: root@host01 /]# ceph config set client.rgw rgw_dns_s3website_name objects-website-zonegroup.example.com
[ceph: root@host01 /]# ceph config set client.rgw rgw_resolve_cname true
```

rgw_enable_static_website 設定は **true** にする必要があります。**rgw_enable_apis** 設定は **s3website** API を有効にする必要があります。**rgw_dns_name** 設定および **rgw_dns_s3website_name** 設定は、完全修飾ドメインを提供する必要があります。サイトで正規の名前拡張子を使用している場合は、**rgw_resolve_cname** オプションを **true** に設定します。



重要

rgw_dns_name および **rgw_dns_s3website_name** の完全修飾ドメイン名は重複 **しないでください**。

4.7.4. 静的 Web ホスティング DNS 設定

以下は、想定される DNS 設定の例です。最初の 2 行は、標準の S3 インターフェイスを使用してゲートウェイインスタンスのドメインを指定し、そ IPv4 アドレスおよび IPv6 アドレスを指しています。3 行目は、正規名の拡張を使用して S3 バケットのワイルドカード CNAME 設定を提供します。4 番目と 5 番目の行は、S3 の Web サイトインターフェイスを使用してゲートウェイインスタンスのドメインを指定し、IPv4 アドレスおよび IPv6 アドレスを参照します。

```
objects-zonegroup.domain.com. IN    A 192.0.2.10
objects-zonegroup.domain.com. IN AAAA 2001:DB8::192:0:2:10
*.objects-zonegroup.domain.com. IN CNAME objects-zonegroup.domain.com.
objects-website-zonegroup.domain.com. IN    A 192.0.2.20
objects-website-zonegroup.domain.com. IN AAAA 2001:DB8::192:0:2:20
```

注記

最初の 2 行にある IP アドレスは、4 番目と 5 行目の IP アドレスとは異なります。

マルチサイト設定で Ceph Object Gateway を使用している場合は、ルーティングソリューションを使用してトラフィックをクライアントに最も近いゲートウェイにルーティングすることを検討してください。

Amazon Web Service (AWS) では、ホスト名に一致するように静的 Web ホストバケットが必要です。Ceph は DNS を設定するいくつかの方法を提供し、**プロキシーに適合する証明書がある場合に HTTPS は機能します**。

サブドメインのバケットのホスト名

AWS 形式の S3 サブドメインを使用するには、DNS エントリーでワイルドカードを使用し、要求を任意のバケットにリダイレクトできます。DNS エントリーは以下のようになります。

```
*.objects-website-zonegroup.domain.com. IN CNAME objects-website-zonegroup.domain.com.
```

以下の方法でバケット名 (バケット名は **bucket1**) にアクセスします。

```
http://bucket1.objects-website-zonegroup.domain.com
```

一致しないバケットのホスト名

Ceph は、リクエストにバケット名を含めずにドメイン名をバケットへのマッピングをサポートします。これは Ceph Object Gateway に固有のものです。ドメイン名を使用してバケットにアクセスするには、ドメイン名をバケット名にマップします。DNS エントリーは以下のようになります。

```
www.example.com. IN CNAME bucket2.objects-website-zonegroup.domain.com.
```

バケット名は **bucket2** です。

以下の方法でバケットにアクセスします。

```
http://www.example.com
```

CNAME を使用した長いバケットのホスト名

AWS は通常、ドメイン名に一致するバケット名を必要とします。CNAME を使用して静的 Web ホストに DNS を設定するには、DNS エントリーは以下のようになります。

```
www.example.com. IN CNAME www.example.com.objects-website-zonegroup.domain.com.
```

以下の方法でバケットにアクセスします。

```
http://www.example.com
```

CNAME のない長いバケットのホスト名

DNS 名には、**SOA**、**NS**、**MX**、**TXT** などの他の非 CNAME レコードが含まれている場合、DNS レコードはドメイン名を IP アドレスに直接マップする必要があります。以下に例を示します。

```
www.example.com. IN A 192.0.2.20
www.example.com. IN AAAA 2001:DB8::192:0:2:20
```

以下の方法でバケットにアクセスします。

```
http://www.example.com
```

4.7.5. 静的 Web ホストサイトの作成

静的 Web サイトを作成するには、以下の手順を実行します。

1. S3 バケットを作成します。バケット名は、Web サイトのドメイン名と同じである場合があります。たとえば、**mysite.com** のバケット名は **mysite.com** になります。これは AWS に必要ですが、Ceph には必要ありません。

   - 詳細は、**Red Hat Ceph Storage オブジェクトゲートウェイ** の **静的 Web ホスティング DNS 設定** セクションを参照してください。

2. 静的 Web コンテンツをバケットにアップロードします。コンテンツには、HTML、CSS、クライアント側の JavaScript、イメージ、音声/ビデオコンテンツなどのダウンロード可能なファイルが含まれる場合があります。Web サイトには **index.html** ファイルが必要で、**error.html** ファイルも必要な場合があります。

3. Web サイトの内容を確認します。この時点で、バケットの作成者のみがコンテンツにアクセスできます。

4. ファイルにパーミッションを設定し、それらが一般に公開されるようにします。

4.8. CEPH OBJECT GATEWAY の高可用性

ストレージ管理者は、Ceph Object Gateway のインスタンス数を単一のゾーンに割り当てることができます。これにより、負荷の増加 (つまり同じゾーングループおよびゾーン) としてスケールアウトすることができます。ただし、高可用性プロキシーを使用するためにフェデレーションされたアーキテクチャーは必要ありません。各 Ceph Object Gateway デーモンには独自の IP アドレスがあるため、**Ingress** サービスを使用して、多数の Ceph Object Gateway デーモンまたはノードで負荷を分散できます。**Ingress** サービスは、Ceph Object Gateway 環境の HAProxy および **keepalived** デーモンを管理します。HAProxy サーバーで HTTPS トラフィックを終了し、HAProxy サーバーと Ceph Object Gateway の Beast フロントエンド Web サーバーインスタンスの間に HTTP を使用することもできます。

前提条件

前提条件

- 異なるホストで実行している 2 つ以上の Ceph Object Gateway デーモン。
- 異なるホストで実行されている **Ingress** サービスの 2 つ以上のインスタンスの容量。

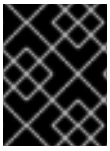
4.8.1. 高可用性サービス

Ingress サービスは、Ceph Object Gateway に可用性の高いエンドポイントを提供します。 **Ingress** サービスは、必要に応じて任意の数のホストにデプロイできます。Red Hat では、サポートされている Red Hat Enterprise Linux サーバーを少なくとも 2 台用意し、各サーバーに **Ingress** サービスを設定することを推奨します。最小限の設定オプションを使用して、高可用性 (HA) サービスを実行できます。Ceph オーケストレーターは、フローティング仮想 IP アドレスで負荷分散を提供することで **haproxy** および **keepalived** デーモンを管理する **ingress** サービスをデプロイします。アクティブな **haproxy** は、利用可能なすべての Ceph Object Gateway デーモンに、すべての Ceph Object Gateway 要求を配布します。

仮想 IP アドレスは、いずれかの **Ingress** ホスト (プライマリーホスト) でまとめて自動的に設定されます。Ceph オーケストレーターは、同じサブネットの一部として設定された既存の IP アドレスに基づいて最初のネットワークインターフェイスを選択します。仮想 IP アドレスが同じサブネットに属さない場合、既存の IP アドレスに一致するように Ceph オーケストレーターのサブネットリストを定義することができます。**keepalived** デーモンとアクティブな **haproxy** がプライマリーホストで応答しない場合、仮想 IP アドレスはバックアップホストに移動します。このバックアップホストが新しいプライマリーホストになります。

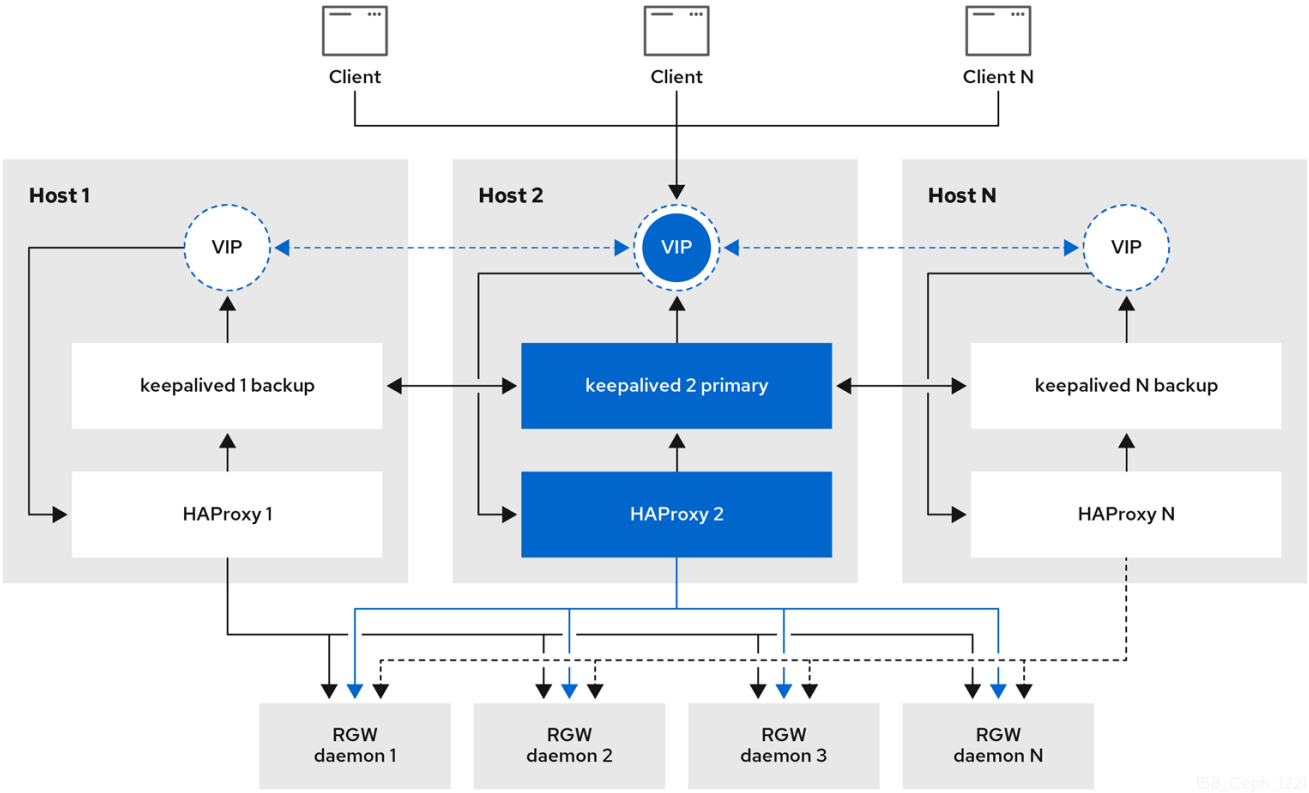
警告

現在、IP アドレスが設定されていないネットワークインターフェイスに仮想 IP アドレスを設定することはできません。



重要

セキュアソケットレイヤー (SSL) を使用するには、Ceph Object Gateway ではなく **Ingress** サービスによって SSL を終了する必要があります。



4.8.2. Ceph Object Gateway の高可用性の設定

Ceph Object Gateway の高可用性 (HA) を設定するには、YAML 設定ファイルを作成します。また、Ceph オーケストレーターが **Ingress** サービスのインストール、設定、および管理を行います。**Ingress** サービスは **haproxy** および **keepalived** デーモンを使用して Ceph Object Gateway の高可用性を提供します。

前提条件

- **Ingress** サービスをインストールするための、Red Hat Enterprise Linux 9 以降を実行する 2 つ以上のホスト。
- 正常かつ実行中の Red Hat Ceph Storage クラスター
- 異なるホストで実行されている 2 つ以上の Ceph Object Gateway デーモン。
- **Ingress** サービスを実行しているホストへのルートレベルのアクセス。
- ファイアウォールを使用している場合は、HTTP 用にポート 80 を開き、HTTPS トラフィック用にポート 443 を開きます。

手順

1. 新規 **ingress.yaml** ファイルを作成します。

   例

   ```
   [root@host01 ~] touch ingress.yaml
   ```

2. **ingress.yaml** ファイルを開いて編集します。以下のオプションを追加し、環境に適した値を追加します。

構文

```
service_type: ingress (1)
service_id: SERVICE_ID (2)
placement: (3)
  hosts:
    - HOST1
    - HOST2
    - HOST3
spec:
  backend_service: SERVICE_ID
  virtual_ip: IP_ADDRESS/CIDR (4)
  frontend_port: INTEGER (5)
  monitor_port: INTEGER (6)
  virtual_interface_networks: (7)
    - IP_ADDRESS/CIDR
  ssl_cert: | (8)
```

(1) **ingress** に設定される必要があります。

(2) 既存の Ceph Object Gateway サービス名と一致する必要があります。

(3) **haproxy** および **keepalived** コンテナーをデプロイする場所。

(4) **ingress** サービスを利用できる仮想 IP アドレス。

(5) **ingress** サービスにアクセスするためのポート。

(6) **haproxy** ロードバランサーのステータスにアクセスするためのポート。

(7) 利用可能なサブネットのオプションリスト。

(8) オプションの SSL 証明書および秘密鍵。

例

```
service_type: ingress
service_id: rgw.foo
placement:
  hosts:
    - host01.example.com
    - host02.example.com
    - host03.example.com
spec:
  backend_service: rgw.foo
  virtual_ip: 192.168.1.2/24
  frontend_port: 8080
  monitor_port: 1967
  virtual_interface_networks:
    - 10.10.0.0/16
  ssl_cert: |
```

```
    -----BEGIN CERTIFICATE-----
    MIIEpAIBAAKCAQEA+Cf4l9OagD6x67HhdCy4Asqw89Zz9ZuGbH50/7ltIMQpJJU0
    gu9ObNtIoC0zabJ7n1jujueYgIpOqGnhRSvsGJiEkgN81NLQ9rqAVaGpadjrNLcM
    bpgqJCZj0vzzmtFBCtenpb5l/EccMFcAydGtGeLP33SaWiZ4Rne56GBInk6SATI/
    JSKweGD1y5GiAWipBR4C74HiAW9q6hCOuSdp/2WQxWT3T1j2sjlqxkHdtInUtwOm
    j5Ism276IndeQ9hR3reFR8PJnKIPx73oTBQ7p9CMR1J4ucq9Ny0J12wQYT00fmJp
    -----END CERTIFICATE-----
    -----BEGIN PRIVATE KEY-----
    MIIEBTCCAu2gAwIBAgIUGfYFsj8HyA9Zv2l600hxzT8+gG4wDQYJKoZIhvcNAQEL

BQAwgYkxCzAJBgNVBAYTAklOMQwwCgYDVQQIDANLQVIxDDAKBgNVBAcMA0JMUjEM

MAoGA1UECgwDUkhUMQswCQYDVQQLDAJCVTEkMCIGA1UEAwwbY2VwaC1zc2wtcmhj
    czUtOGRjeHY2LW5vZGU1MR0wGwYJKoZIhvcNAQkBFg5hYmNAcmVkaGF0LmNvbTAe
    -----END PRIVATE KEY-----
```

3. Cephadm シェルを起動します。

   例

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

4. 最新の **haproxy** イメージおよび **keepalived** イメージを設定します。

   構文

```
ceph config set mgr mgr/cephadm/container_image_haproxy HAPROXY_IMAGE_ID
ceph config set mgr mgr/cephadm/container_image_keepalived KEEPALIVED_IMAGE_ID
```

   Red Hat Enterprise Linux 9

```
[ceph: root@host01 /]# ceph config set mgr mgr/cephadm/container_image_haproxy
registry.redhat.io/rhceph/rhceph-haproxy-rhel9:latest
[ceph: root@host01 /]# ceph config set mgr mgr/cephadm/container_image_keepalived
registry.redhat.io/rhceph/keepalived-rhel9:latest
```

5. Ceph オーケストレーターを使用して新規 **Ingress** サービスをインストールおよび設定します。

```
[ceph: root@host01 /]# ceph orch apply -i /var/lib/ceph/radosgw/ingress.yaml
```

6. Ceph オーケストレーターが完了したら、HA 設定を確認します。

   a. **Ingress** サービスを実行しているホストで、仮想 IP アドレスが表示されることを確認します。

      例

```
[root@host01 ~]# ip addr show
```

   b. Ceph クライアントから Ceph Object Gateway にアクセスしてみてください。

      構文

```
wget HOST_NAME
```

例

```
[root@client ~]# wget host01.example.com
```

以下の例のような内容を含む **index.html** が返される場合、Ceph Object Gateway の HA 設定は正常に機能しています。

例

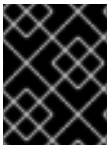
```
<?xml version="1.0" encoding="UTF-8"?>
 <ListAllMyBucketsResult xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
  <Owner>
   <ID>anonymous</ID>
   <DisplayName></DisplayName>
  </Owner>
  <Buckets>
  </Buckets>
 </ListAllMyBucketsResult>
```

関連情報

- 詳細は、**標準的な RHEL インストールの実行ガイド** を参照してください。
- 詳細は、**Red Hat Ceph Storage オブジェクトゲートウェイガイド** の **高可用性サービス** セクションを参照してください。

4.8.3. NFS-Ganesha への名前空間のエクスポート

Ceph Object Gateway で使用する新たな NFS Ganesha エクスポートを設定するには、Red Hat Ceph Storage Dashboard を使用する必要があります。詳細は、**Red Hat Ceph Storage ダッシュボードガイド** の **Ceph Dashboard の NFS Ganesha エクスポートの管理**セクションを参照してください。



重要

Ceph Object Gateway を使用する既存の NFS 環境では、現時点で Red Hat Ceph Storage 4 から Red Hat Ceph Storage 5 へのアップグレードはサポートされません。

重要

Red Hat は、Ceph Object Gateway を使用した NFS バージョン 3 エクスポートをサポートしません。

第5章 マルチサイト設定および管理

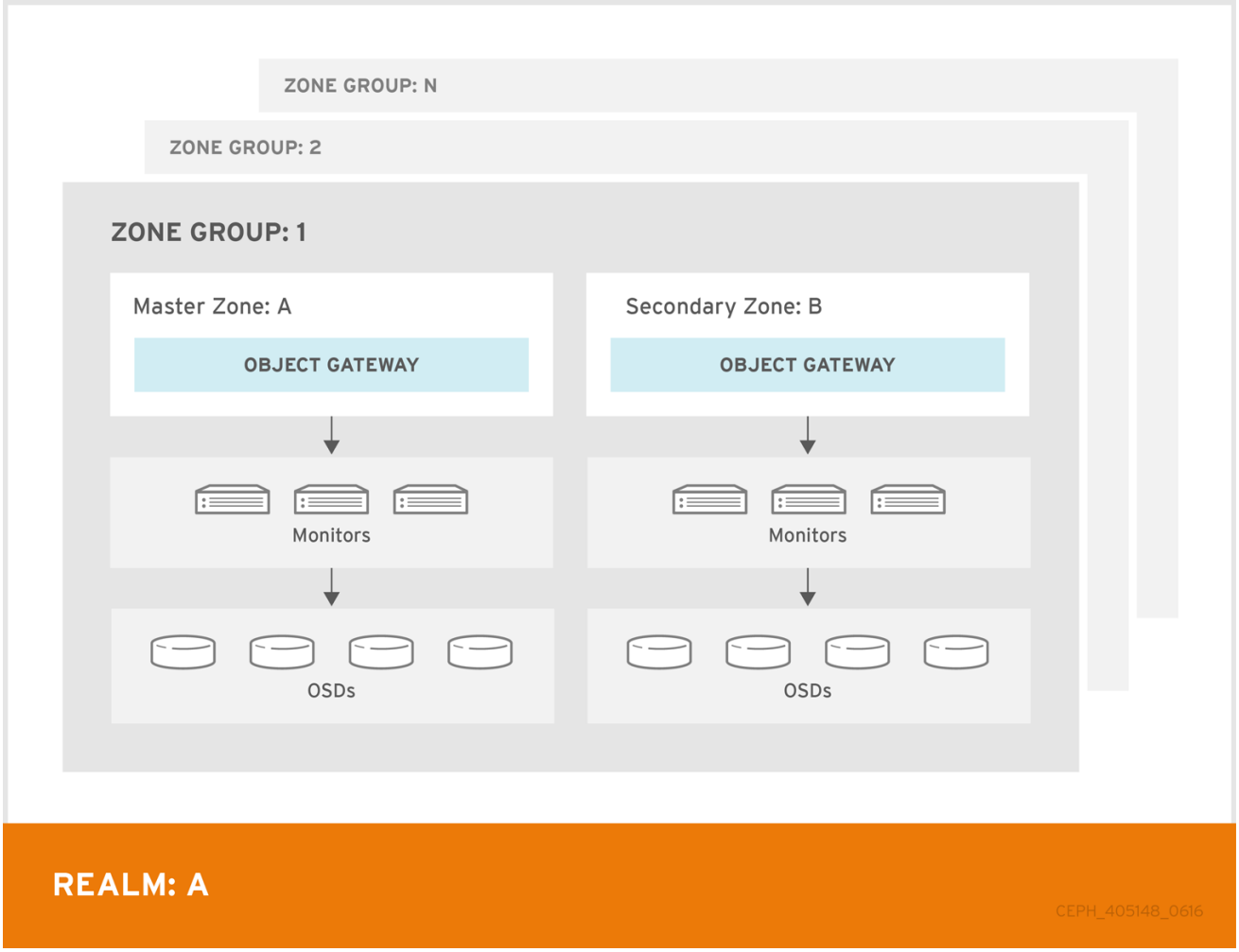
ストレージ管理者は、さまざまなユースケースに対して複数の Ceph Object Gateway を設定し、管理することができます。障害復旧中の手順、およびフェイルオーバーイベントを確認できます。また、マルチサイトの Ceph Object Gateway 環境でレルム、ゾーン、および同期ポリシーについて詳しく説明します。

単一ゾーン設定は通常、1つのゾーンと1つ以上の **ceph-radosgw** インスタンスを含む1つのゾーングループで設定され、インスタンス間でゲートウェイクライアント要求の負荷を分散できます。単一ゾーン設定では、通常複数のゲートウェイインスタンスが単一の Ceph Storage Cluster を参照します。ただし、Red Hat は、Ceph Object Gateway の複数のマルチサイト設定オプションをサポートしています。

- **マルチゾーン:** より高度な設定は、1つのゾーングループと複数のゾーンで設定され、各ゾーンには1つ以上の **ceph-radosgw** インスタンスがあります。各ゾーンは、独自の Ceph Storage Cluster でサポートされます。ゾーングループ内の複数のゾーンは、ゾーンの1つで重大な障害が発生した場合に、ゾーングループに障害復旧を提供します。各ゾーンはアクティブで、書き込み操作を受け取る可能性があります。障害復旧に加え、複数のアクティブなゾーンがコンテンツ配信ネットワークの基盤としても機能する場合があります。レプリケーションなしで複数のゾーンを設定するには、**レプリケーションなしで複数のゾーンを設定** を参照してください。

- **マルチゾーングループ:** 以前はリージョンと呼ばれていた Ceph Object Gateway は、複数のゾーングループをサポートすることもでき、各ゾーングループには1つ以上のゾーンがあります。同じレルム内のゾーングループに保管されるオブジェクトは、グローバル名前空間を共有し、ゾーングループとゾーン全体で一意のオブジェクト ID を確保します。

- **複数のレルム:** Ceph Object Gateway は、レルムの概念をサポートします。レルムは、単一のゾーングループまたは複数のゾーングループと、レルムのグローバルに一意の名前空間です。複数のレルムは、多数の設定と名前空間をサポートする機能を提供します。



前提条件

- 正常かつ実行中の Red Hat Ceph Storage クラスター
- Ceph Object Gateway ソフトウェアの デプロイメント。

5.1. 要件および前提条件

マルチサイト設定には、少なくとも 2 つの Ceph Storage Cluster が必要です。さらに、2 つ以上の Ceph Object Gateway インスタンス (Ceph Storage Cluster ごとに 1 つずつ) が必要です。

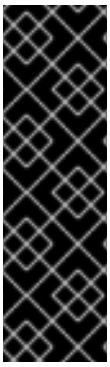
本ガイドでは、地理的に別々の場所にある Ceph Storage Cluster が 2 つ以上あることを前提としていますが、この設定は同じ物理サイトで機能することができます。。本ガイドでは、**rgw1**、**rgw2**、**rgw3**、および **rgw4** という名前の 4 つの Ceph Object Gateway サーバーをそれぞれ前提としています。

マルチサイト設定では、マスターゾーングループとマスターゾーンが必要です。さらに、**各ゾーングループにはマスターゾーンが必要です**。ゾーングループには、1 つ以上のセカンダリーゾーンまたはマスター以外のゾーンがあります。

重要

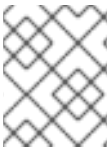
マルチサイトのネットワークに関する考慮事項を計画するときは、マルチサイト同期ネットワークで観察される帯域幅と遅延の関係、およびセカンダリーサイトに負っているオブジェクトの現在の同期状態と直接相関するクライアント取り込み速度を理解することが重要です。Red Hat Ceph Storage マルチサイトクラスター間のネットワークリンクは、プライマリークラスターへの取り込みを処理して、セカンダリーサイトで効果的な復旧時間を維持できる必要があります。マルチサイト同期は非同期であり、制限の 1 つは、同期ゲートウェイがリンクを介してデータを処理できる速度です。ネットワークの相互接続速度に関して検討する例としては、クライアントゲートウェイごとに 8 TB または累積受信データごとに 1 GbE またはデータセンター間接続が考えられます。したがって、他の 2 つのサイトにレプリケートし、1 日 16 TB を取り込む場合、マルチサイトレプリケーションには 6 GbE の専用帯域幅が必要です。

Red Hat では、追加のオーバーヘッドが発生するため、インターネット経由の VPN は理想的ではないため、プライベートイーサネットまたは高密度波長分割多重 (DWDM) も推奨しています。



重要

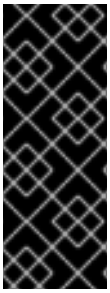
レルムのマスターゾーングループ内のマスターゾーンは、(**radosgw-admin** CLI によって作成された) ユーザー、クォータ、バケットを含むレルムのメタデータのマスターコピーを保存するロールを果たします。このメタデータは、セカンダリーゾーンおよびセカンダリーゾーングループに自動的に同期されます。**radosgw-admin** CLI で実行されるメタデータ操作は、セカンダリーゾーングループおよびゾーンに確実に同期されるように、マスターゾーングループのマスターゾーン内のホストで **実行する必要** があります。現在、セカンダリーゾーンおよびゾーングループでメタデータ操作を実行することは **可能** ですが、それらが同期**されず**、メタデータが断片化されるため、推奨**されません**。



注記

新しい Ceph Object Gateway をマルチサイトにデプロイメントする場合、メタデータ操作をセカンダリーサイトに同期するには約 20 分かかります。

次の例では、**rgw1** ホストがマスターゾーングループのマスターゾーンとして機能します。**rgw2** ホストは、マスターゾーングループのセカンダリーゾーンとして機能します。**rgw3** ホストは、セカンダリーゾーングループのマスターゾーンとして機能します。**rgw4** ホストは、セカンダリーゾーングループのセカンダリーゾーンとして機能します。



重要

Red Hat では、ロードバランサーと 3 つの Ceph Object Gateway デーモンを使用して、エンドポイントをマルチサイトと同期させることを推奨します。マルチサイト設定の非同期 Ceph Object Gateway ノード (ロードバランサーを介したクライアント I/O 操作専用) の場合は、**ceph config set client.rgw.CLIENT_NODE rgw_run_sync_thread false** コマンドを実行して、同期操作が実行されないようにします。その後、Ceph Object Gateway を再起動します。

以下は、ゲートウェイを同期するための HAProxy の一般的な設定ファイルです。

例

```
[root@host01 ~]# cat ./haproxy.cfg
```

```
global

 log      127.0.0.1 local2

 chroot   /var/lib/haproxy
 pidfile  /var/run/haproxy.pid
 maxconn  7000
 user     haproxy
 group    haproxy
 daemon

 stats socket /var/lib/haproxy/stats

defaults

 mode                    http
 log                     global
 option                  httplog
 option                  dontlognull
 option http-server-close
 option forwardfor    except 127.0.0.0/8
 option                  redispatch
 retries                 3
 timeout http-request 10s
 timeout queue           1m
 timeout connect         10s
 timeout client          30s
 timeout server          30s
 timeout http-keep-alive 10s
 timeout check           10s
        timeout client-fin 1s
        timeout server-fin 1s
 maxconn                 6000


listen  stats
 bind 0.0.0.0:1936
    mode         http
    log          global

    maxconn 256


    clitimeout   10m
    srvtimeout   10m
    contimeout   10m
    timeout queue   10m

# JTH start
    stats enable
    stats hide-version
    stats refresh 30s
    stats show-node
##     stats auth admin:password
    stats uri  /haproxy?stats
      stats admin if TRUE
```

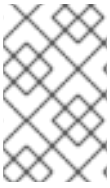
```
frontend  main
 bind  *:5000
 acl url_static    path_beg    -i /static /images /javascript /stylesheets
 acl url_static    path_end    -i .jpg .gif .png .css .js

 use_backend static       if url_static
 default_backend          app
       maxconn 6000

backend static
 balance  roundrobin
   fullconn 6000
   server  app8 host01:8080 check maxconn 2000
   server  app9 host02:8080 check maxconn 2000
   server  app10 host03:8080 check maxconn 2000


backend app
 balance  roundrobin
   fullconn 6000
   server  app8 host01:8080 check maxconn 2000
   server  app9 host02:8080 check maxconn 2000
   server  app10 host03:8080 check maxconn 2000
```

5.2. POOLS

Red Hat は、**Ceph 配置グループのプールごとの計算機** を使用して、**radosgw** デーモンが作成するプールに適した配置グループの数を計算することを推奨します。Ceph 設定データベースで計算された値をデフォルト値として設定します。

例

```
[ceph: root@host01 /]# ceph config set osd osd_pool_default_pg_num 50
[ceph: root@host01 /]# ceph config set osd osd_pool_default_pgp_num 50
```



注記

Ceph の設定にこの変更を行うと、Ceph Object Gateway インスタンスがプールを作成する際にこれらのデフォルトが使用されます。または、プールを手動で作成することもできます。

ゾーンに固有のプール名は、命名規則 **ZONE_NAME.POOL_NAME** に従います。たとえば、**us-east** という名前のゾーンには以下のプールがあります。

- **.rgw.root**
- **us-east.rgw.control**
- **us-east.rgw.meta**
- **us-east.rgw.log**

- **us-east.rgw.buckets.index**
- **us-east.rgw.buckets.data**
- **us-east.rgw.buckets.non-ec**
- **us-east.rgw.meta:users.keys**
- **us-east.rgw.meta:users.email**
- **us-east.rgw.meta:users.swift**
- **us-east.rgw.meta:users.uid**

関連情報

- プールの作成に関する詳細は、**Red Hat Ceph Storage ストラテジーガイド**の プール の章を参照してください。

5.3. シングルサイトシステムからマルチサイトへの移行

default ゾーングループとゾーンを持つシングルサイトシステムからマルチサイトシステムに移行するには、次の手順を使用します。

1. レルムを作成します。**NAME** を、レルム名に置き換えます。

   構文

   ```
   radosgw-admin realm create --rgw-realm=NAME --default
   ```

2. デフォルトゾーンとゾーングループの名前を変更します。**<name>** を、ゾーングループまたはゾーン名に置き換えます。

   構文

   ```
   radosgw-admin zonegroup rename --rgw-zonegroup default --zonegroup-new-name=NEW_ZONE_GROUP_NAME
   radosgw-admin zone rename --rgw-zone default --zone-new-name us-east-1 --rgw-zonegroup=ZONE_GROUP_NAME
   ```

3. プライマリーゾーングループを設定します。**NAME** を、レルムまたはゾーングループ名に置き換えます。**FQDN** を、ゾーングループの完全修飾ドメイン名に置き換えます。

   構文

   ```
   radosgw-admin zonegroup modify --rgw-realm=REALM_NAME --rgw-zonegroup=ZONE_GROUP_NAME --endpoints http://FQDN:80 --master --default
   ```

4. システムユーザーを作成します。**USER_ID** を、ユーザー名に置き換えます。**DISPLAY_NAME** を、表示名に置き換えます。これにはスペースを含めることができます。

   構文

```
radosgw-admin user create --uid=USER_ID \
                  --display-name="DISPLAY_NAME" \
                  --access-key=ACCESS_KEY --secret=SECRET_KEY \ --system
```

5. プライマリーゾーンを設定します。**NAME** を、レルム、ゾーングループ、またはゾーン名に置き換えます。**FQDN** を、ゾーングループの完全修飾ドメイン名に置き換えます。

   構文

```
radosgw-admin zone modify --rgw-realm=REALM_NAME --rgw-
zonegroup=ZONE_GROUP_NAME \
                  --rgw-zone=ZONE_NAME --endpoints http://FQDN:80 \
                  --access-key=ACCESS_KEY --secret=SECRET_KEY \
                  --master --default
```

6. Ceph 設定データベースを更新します。

   構文

```
ceph config set client.rgw.SERVICE_NAME rgw_realm REALM_NAME
ceph config set client.rgw.SERVICE_NAME rgw_zonegroup ZONE_GROUP_NAME
ceph config set client.rgw.SERVICE_NAME rgw_zone PRIMARY_ZONE_NAME
```

   例

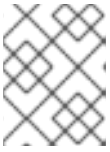
```
[ceph: root@host01 /]# ceph config set client.rgw.rgwsvcid.mons-1.jwgwwp rgw_realm
test_realm
[ceph: root@host01 /]# ceph config set client.rgw.rgwsvcid.mons-1.jwgwwp rgw_zonegroup
us
[ceph: root@host01 /]# ceph config set client.rgw.rgwsvcid.mons-1.jwgwwp rgw_zone us-
east-1
```

7. 更新された設定をコミットします。

   例

```
[ceph: root@host01 /]# radosgw-admin period update --commit
```

8. Ceph Object Gateway を再起動します。



注記

NAME 列の **ceph orch ps** コマンドの出力を使用して、SERVICE_TYPE.ID 情報を取得します。

   a. ストレージクラスター内の個別のノードで Ceph Object Gateway を再起動するには、以下を実行します。

      構文

```
systemctl restart ceph-CLUSTER_ID@SERVICE_TYPE.ID.service
```

例

```
[root@host01 ~]# systemctl restart ceph-c4b34c6f-8365-11ba-dc31-529020a7702d@rgw.realm.zone.host01.gwasto.service
```

b. ストレージクラスター内のすべてのノードで Ceph Object Gateway を再起動するには、以下を実行します。

構文

```
ceph orch restart SERVICE_TYPE
```

例

```
[ceph: root@host01 /]# ceph orch restart rgw
```

9. セカンダリーゾーンを確立します。Establishing a secondary zone セクションを参照してください。

5.4. セカンダリーゾーンの確立

ゾーングループ内のゾーンは、すべてのデータを複製して、各ゾーンが同じデータを持つようにします。セカンダリーゾーンを作成するときは、セカンダリーゾーンにサービスを提供するように識別されたホストで **すべて** の **radosgw-admin zone** 操作を発行します。

注記

ゾーンを追加するには、セカンダリーゾーンを追加する手順と同じ手順に従います。別のゾーン名を使用します。

重要

マスターゾーングループのマスターゾーン内のホストで、ユーザーの作成やクォータなどのメタデータ操作を実行する必要があります。マスターゾーンおよびセカンダリーゾーンは、RESTful API からバケット操作を受信できますが、セカンダリーゾーンはバケット操作をマスターゾーンにリダイレクトします。マスターゾーンがダウンしている場合、バケット操作は失敗します。**radosgw-admin** CLI を使用してバケットを作成する場合は、バケットが他のゾーングループおよびゾーンと同期するように、マスターゾーングループのマスターゾーン内のホストでバケットを実行する必要があります。

前提条件

- 少なくとも 2 つの実行中の Red Hat Ceph Storage クラスター
- 少なくとも 2 つの Ceph Object Gateway インスタンス (各 Red Hat Ceph Storage クラスターに 1 つ)。
- すべてのノードへの root レベルのアクセス。
- ノードまたはコンテナーがストレージクラスターに追加されます。
- すべての Ceph Manager、Monitor、および OSD デーモンがデプロイされます。

手順

1. **cephadm** シェルにログインします。

   例

   ```
   [root@host04 ~]# cephadm shell
   ```

2. ホストからプライマリーレルム設定をプルします。

   構文

   ```
   radosgw-admin realm pull --url=URL_TO_PRIMARY_ZONE_GATEWAY --access-key=ACCESS_KEY --secret-key=SECRET_KEY
   ```

   例

   ```
   [ceph: root@host04 /]# radosgw-admin realm pull --url=http://10.74.249.26:80 --access-key=LIPEYZJLTWXRKXS9LPJC --secret-key=IsAje0AVDNXNw48LjMAimpCpI7VaxJYSnfD0FFKQ
   ```

3. ホストからプライマリー期間設定をプルします。

   構文

   ```
   radosgw-admin period pull --url=URL_TO_PRIMARY_ZONE_GATEWAY --access-key=ACCESS_KEY --secret-key=SECRET_KEY
   ```

   例

   ```
   [ceph: root@host04 /]# radosgw-admin period pull --url=http://10.74.249.26:80 --access-key=LIPEYZJLTWXRKXS9LPJC --secret-key=IsAje0AVDNXNw48LjMAimpCpI7VaxJYSnfD0FFKQ
   ```

4. セカンダリーゾーンを設定します。

   注記

   デフォルトでは、すべてのゾーンはアクティブ/アクティブ設定で実行されます。つまり、ゲートウェイクライアントは任意のゾーンにデータを書き込むことができ、ゾーンはゾーングループ内の他のすべてのゾーンにデータを複製します。セカンダリーゾーンが書き込み操作を受け入れない場合は、**--read-only** フラグを指定して、マスターゾーンとセカンダリーゾーンの間にアクティブ-パッシブ設定を作成します。さらに、マスターゾーングループのマスターゾーンに格納されている、生成されたシステムユーザーの **access_key** および **secret_key** を指定します。

   構文

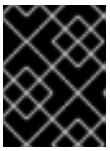
   ```
   radosgw-admin zone create --rgw-zonegroup=_ZONE_GROUP_NAME_ \
             --rgw-zone=_SECONDARY_ZONE_NAME_ --
   endpoints=http://_RGW_SECONDARY_HOSTNAME_:_RGW_PRIMARY_PORT_NUMBER_
   ```

```
1_ \
        --access-key=_SYSTEM_ACCESS_KEY_ --secret=_SYSTEM_SECRET_KEY_ \
        [--read-only]
```

例

```
[ceph: root@host04 /]# radosgw-admin zone create --rgw-zonegroup=us --rgw-zone=us-east-2 --endpoints=http://rgw2:80 --access-key=LIPEYZJLTWXRKXS9LPJC --secret-key=IsAje0AVDNXNw48LjMAimpCpI7VaxJYSnfD0FFKQ
```

5. 必要に応じて、デフォルトゾーンを削除します。



重要

デフォルトゾーンおよびゾーングループを使用してデータを保存する場合は、デフォルトゾーンとそのプールは削除しないでください。

例

```
[ceph: root@host04 /]# radosgw-admin zone rm --rgw-zone=default
[ceph: root@host04 /]# ceph osd pool rm default.rgw.log default.rgw.log --yes-i-really-really-mean-it
[ceph: root@host04 /]# ceph osd pool rm default.rgw.meta default.rgw.meta --yes-i-really-really-mean-it
[ceph: root@host04 /]# ceph osd pool rm default.rgw.control default.rgw.control --yes-i-really-really-mean-it
[ceph: root@host04 /]# ceph osd pool rm default.rgw.data.root default.rgw.data.root --yes-i-really-really-mean-it
[ceph: root@host04 /]# ceph osd pool rm default.rgw.gc default.rgw.gc --yes-i-really-really-mean-it
```

6. Ceph 設定データベースを更新します。

構文

```
ceph config set client.rgw.SERVICE_NAME rgw_realm REALM_NAME
ceph config set client.rgw.SERVICE_NAME rgw_zonegroup ZONE_GROUP_NAME
ceph config set client.rgw.SERVICE_NAME rgw_zone SECONDARY_ZONE_NAME
```

例

```
[ceph: root@host04 /]# ceph config set client.rgw.rgwsvcid.mons-1.jwgwwp rgw_realm test_realm
[ceph: root@host04 /]# ceph config set client.rgw.rgwsvcid.mons-1.jwgwwp rgw_zonegroup us
[ceph: root@host04 /]# ceph config set client.rgw.rgwsvcid.mons-1.jwgwwp rgw_zone us-east-2
```

7. 変更をコミットします。

構文

```
radosgw-admin period update --commit
```

例

```
[ceph: root@host04 /]# radosgw-admin period update --commit
```

8. **cephadm** シェルの外部で、ストレージクラスターおよびプロセスの FSID を取得します。

   例

   ```
   [root@host04 ~]#  systemctl list-units | grep ceph
   ```

9. Ceph Object Gateway デーモンを起動します。

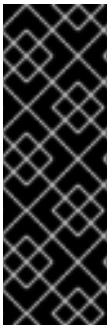
   構文

   ```
   systemctl start ceph-FSID@DAEMON_NAME
   systemctl enable ceph-FSID@DAEMON_NAME
   ```

   例

   ```
   [root@host04 ~]# systemctl start ceph-62a081a6-88aa-11eb-a367-
   001a4a000672@rgw.test_realm.us-east-2.host04.ahdtsw.service
   [root@host04 ~]# systemctl enable ceph-62a081a6-88aa-11eb-a367-
   001a4a000672@rgw.test_realm.us-east-2.host04.ahdtsw.service
   ```

5.5. アーカイブゾーンの設定 (テクノロジープレビュー)



重要

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

Object Storage Archive Zone 機能を使用して、Red Hat Ceph Storage に存在するオブジェクトデータをアーカイブします。

アーカイブゾーンは、Ceph Object Gateway のマルチサイトレプリケーションと S3 オブジェクトのバージョン管理機能を使用します。アーカイブゾーンには、運用ファイルで削除された場合でも、使用可能なすべてのオブジェクトのすべてのバージョンが保持されます。

アーカイブゾーンには、アーカイブゾーンに関連付けられたゲートウェイを介してのみ削除できる S3 オブジェクトのバージョンの履歴があります。すべてのデータ更新およびメタデータを取得し、それらを S3 オブジェクトのバージョンとして統合します。

アーカイブゾーンの作成後には、アーカイブゾーンへのバケットの詳細なレプリケーションを使用できます。

バケットのライフサイクルポリシーを通じてアーカイブゾーンのストレージスペースの使用量を制御でき、オブジェクトに対して保持するバージョンの数を定義できます。

アーカイブゾーンは、論理的または物理的なエラーからデータを保護するのに役立ちます。これにより、実稼働ゾーンでバケットを誤って削除するなどの論理障害からユーザーを守ることができます。また、実稼働サイトの完全な障害など、大規模なハードウェア障害からデータを保存することもできます。さらに、不変のコピーも提供されるため、ランサムウェア保護戦略の構築に役立ちます。

バケットの詳細なレプリケーションを実装するには、ポリシーを有効または無効にする sync policies コマンドを使用します。詳細は **同期ポリシーグループの作成** および **同期ポリシーグループの変更** を参照してください。

注記

同期ポリシーグループ手順の使用はオプションであり、バケットの詳細なレプリケーションで有効化と無効化を使用する場合にのみ必要です。バケットの詳細なレプリケーションを使用せずにアーカイブゾーンを使用する場合、同期ポリシー手順を使用する必要はありません。

ストレージクラスターをシングルサイトから移行する場合は、**シングルサイトシステムのマルチサイトへの移行** を参照してください。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- Ceph Monitor ノードへの root レベルのアクセス。
- Ceph Object Gateway ソフトウェアのインストール。

手順

- 新しいゾーンの作成中に、**archive** 層を使用してアーカイブゾーンを設定します。

  構文

  ```
  radosgw-admin zone create --rgw-zonegroup={ZONE_GROUP_NAME} --rgw-zone={ZONE_NAME} --endpoints={http://FQDN:PORT},{http://FQDN:PORT} --tier-type=archive
  ```

  例

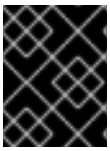
  ```
  [ceph: root@host01 /]# radosgw-admin zone create --rgw-zonegroup=us --rgw-zone=us-east --endpoints={http://example.com:8080} --tier-type=archive
  ```

関連情報

- 詳細は、**Red Hat Ceph Storage Object Gateway ガイド**の **Ceph Orchestrator を使用したマルチサイト Ceph Object Gateway のデプロイ** セクションを参照してください。

5.5.1. アーカイブゾーン内のオブジェクトの削除

S3 ライフサイクルポリシー拡張機能を使用して、**<ArchiveZone>** 要素内のオブジェクトを削除できます。



重要

アーカイブゾーンオブジェクトは、**expiration** ライフサイクルポリシールールを使用してのみ削除できます。

- いずれかの **<Rule>** セクションに **<ArchiveZone>** 要素が含まれている場合、そのルールはアーカイブゾーンで実行され、アーカイブゾーンで実行される唯一のルールになります。
- **<ArchiveZone>** とマークされたルールは、非アーカイブゾーンでは実行されません。

ライフサイクルポリシー内のルールにより、いつ、どのオブジェクトを削除するかが決まります。ライフサイクルの作成と管理の詳細は、**バケットのライフサイクル** を参照してください。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- Ceph Monitor ノードへの root レベルのアクセス。
- Ceph Object Gateway ソフトウェアのインストール。

手順

1. **<ArchiveZone>** ライフサイクルポリシールールを設定します。ライフサイクルポリシーの作成の詳細は、**Red Hat Ceph Storage オブジェクトゲートウェイガイド** の **ライフサイクル管理ポリシーの作成** セクションを参照してください。

   例

   ```
   <?xml version="1.0" ?>
   <LifecycleConfiguration xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
       <Rule>
           <ID>delete-1-days-az</ID>
           <Filter>
      <Prefix></Prefix>
      <ArchiveZone /> (1)
           </Filter>
           <Status>Enabled</Status>
           <Expiration>
               <Days>1</Days>
           </Expiration>
       </Rule>
   </LifecycleConfiguration>
   ```

2. オプション: 特定のライフサイクルポリシーにアーカイブゾーンルールが含まれているかどうかを確認します。

   構文

   ```
   radosgw-admin lc get --bucket BUCKET_NAME
   ```

   例

   ```
   [ceph: root@host01 /]# radosgw-admin lc get --bucket test-bkt
   ```

```
{
    "prefix_map": {
        "": {
            "status": true,
            "dm_expiration": true,
            "expiration": 0,
            "noncur_expiration": 2,
            "mp_expiration": 0,
            "transitions": {},
            "noncur_transitions": {}
        }
    },
    "rule_map": [
        {
            "id": "Rule 1",
            "rule": {
                "id": "Rule 1",
                "prefix": "",
                "status": "Enabled",
                "expiration": {
                    "days": "",
                    "date": ""
                },
                "noncur_expiration": {
                    "days": "2",
                    "date": ""
                },
                "mp_expiration": {
                    "days": "",
                    "date": ""
                },
                "filter": {
                    "prefix": "",
                    "obj_tags": {
                        "tagset": {}
                    },
                    "archivezone": ""  (1)
                },
                "transitions": {},
                "noncur_transitions": {},
                "dm_expiration": true
            }
        }
    ]
}
```

(1) (1) アーカイブゾーンルール。これは、アーカイブゾーンルールを使用したライフサイクルポリシーの例です。

3. Ceph Object Gateway ユーザーが削除されると、そのユーザーが所有するアーカイブサイトのバケットにアクセスできなくなります。これらのバケットを別の Ceph Object Gateway ユーザーにリンクして、データにアクセスします。

   構文

```
radosgw-admin bucket link --uid NEW_USER_ID --bucket BUCKET_NAME --yes-i-really-mean-it
```

例

```
[ceph: root@host01 /]# radosgw-admin bucket link --uid arcuser1 --bucket arc1-deleted-da473fbbaded232dc5d1e434675c1068 --yes-i-really-mean-it
```

関連情報

- 詳細は、**Red Hat Ceph Storage オブジェクトゲートウェイガイド** の バケットライフサイクル セクションを参照してください。
- より詳細は、**Red Hat Ceph Storage 開発者ガイド** の S3 バケットライフサイクル セクションを参照してください。

5.6. フェイルオーバーおよび障害復旧

プライマリーゾーンに障害が発生した場合は、障害復旧のためにセカンダリーゾーンにフェイルオーバーします。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- Ceph Monitor ノードへの root レベルのアクセス。
- Ceph Object Gateway ソフトウェアのインストール。

手順

1. セカンダリーゾーンをプライマリーおよびデフォルトゾーンにします。以下に例を示します。

   構文

   ```
   radosgw-admin zone modify --rgw-zone=ZONE_NAME --master --default
   ```

   デフォルトでは、Ceph Object Gateway は active-active 設定で実行されます。クラスターが active-passive 設定で実行されるように設定されている場合、セカンダリーゾーンは読み取り専用ゾーンになります。ゾーンが書き込み操作を受け取れるように **--read-only** ステータスを削除します。以下に例を示します。

   構文

   ```
   radosgw-admin zone modify --rgw-zone=ZONE_NAME --master --default --read-only=false
   ```

2. 期間を更新して、変更を反映します。

   例

   ```
   [ceph: root@host01 /]# radosgw-admin period update --commit
   ```

3. Ceph Object Gateway を再起動します。

注記

NAME 列の **ceph orch ps** コマンドの出力を使用して、**SERVICE_TYPE.ID** 情報を取得します。

a. ストレージクラスター内の個別のノードで Ceph Object Gateway を再起動するには、以下を実行します。

構文

```
systemctl restart ceph-CLUSTER_ID@SERVICE_TYPE.ID.service
```

例

```
[root@host01 ~]# systemctl restart ceph-c4b34c6f-8365-11ba-dc31-529020a7702d@rgw.realm.zone.host01.gwasto.service
```

b. ストレージクラスター内のすべてのノードで Ceph Object Gateway を再起動するには、以下を実行します。

構文

```
ceph orch restart SERVICE_TYPE
```

例

```
[ceph: root@host01 /]# ceph orch restart rgw
```

以前のプライマリーゾーンが復旧する場合は、操作を元に戻します。

1. 復旧したゾーンから、現在のプライマリーゾーンからレルムをプルします。

構文

```
radosgw-admin realm pull --url=URL_TO_PRIMARY_ZONE_GATEWAY \
                         --access-key=ACCESS_KEY --secret=SECRET_KEY
```

2. 復旧ゾーンをプライマリーおよびデフォルトゾーンにします。

構文

```
radosgw-admin zone modify --rgw-zone=ZONE_NAME --master --default
```

3. 期間を更新して、変更を反映します。

例

```
[ceph: root@host01 /]# radosgw-admin period update --commit
```

4. 復旧されたゾーンで Ceph Object Gateway を再起動します。

   構文

   ```
   ceph orch restart SERVICE_TYPE
   ```

   例

   ```
   [ceph: root@host01 /]# ceph orch restart rgw
   ```

5. セカンダリーゾーンを読み取り専用設定を使用する必要がある場合は、セカンダリーゾーンを更新します。

   構文

   ```
   radosgw-admin zone modify --rgw-zone=ZONE_NAME --read-only
   radosgw-admin zone modify --rgw-zone=ZONE_NAME --read-only
   ```

6. 期間を更新して、変更を反映します。

   例

   ```
   [ceph: root@host01 /]# radosgw-admin period update --commit
   ```

7. セカンダリーゾーンで Ceph Object Gateway を再起動します。

   構文

   ```
   ceph orch restart SERVICE_TYPE
   ```

   例

   ```
   [ceph: root@host01 /]# ceph orch restart rgw
   ```

5.7. レプリケーションなしで複数のゾーンを設定

互いをレプリケートしない複数のゾーンを設定できます。たとえば、会社内の各チームに専用のゾーンを作成できます。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- Ceph Object Gateway ソフトウェアのインストール。
- Ceph Object Gateway ノードへのルートレベルのアクセスがある。

手順

1. レルムを新規作成します。

   構文

```
radosgw-admin realm create --rgw-realm=REALM_NAME [--default]
```

例

```
[ceph: root@host01 /]# radosgw-admin realm create --rgw-realm=test_realm --default
{
    "id": "0956b174-fe14-4f97-8b50-bb7ec5e1cf62",
    "name": "test_realm",
    "current_period": "1950b710-3e63-4c41-a19e-46a715000980",
    "epoch": 1
}
```

2. 新しいゾーングループを作成します。

構文

```
radosgw-admin zonegroup create --rgw-zonegroup=ZONE_GROUP_NAME --
endpoints=FQDN:PORT --rgw-realm=REALM_NAME|--realm-id=REALM_ID --master --
default
```

例

```
[ceph: root@host01 /]# radosgw-admin zonegroup create --rgw-zonegroup=us --
endpoints=http://rgw1:80 --rgw-realm=test_realm --master --default
{
    "id": "f1a233f5-c354-4107-b36c-df66126475a6",
    "name": "us",
    "api_name": "us",
    "is_master": "true",
    "endpoints": [
        "http:\/\/rgw1:80"
    ],
    "hostnames": [],
    "hostnames_s3webzone": [],
    "master_zone": "",
    "zones": [],
    "placement_targets": [],
    "default_placement": "",
    "realm_id": "0956b174-fe14-4f97-8b50-bb7ec5e1cf62"
}
```

3. ユースケースに応じて、1 つ以上のゾーンを作成します。

構文

```
radosgw-admin zone create --rgw-zonegroup=ZONE_GROUP_NAME --rgw-
zone=ZONE_NAME --master --default --endpoints=FQDN:PORT,FQDN:PORT
```

例

```
[ceph: root@host01 /]# radosgw-admin zone create --rgw-zonegroup=us --rgw-zone=us-east
--master --default --endpoints=http://rgw1:80
```

4. ゾーングループの設定が含まれる JSON ファイルを取得します。

構文

```
radosgw-admin zonegroup get --rgw-zonegroup=ZONE_GROUP_NAME > JSON_FILE_NAME
```

例

```
[ceph: root@host01 /]# radosgw-admin zonegroup get --rgw-zonegroup=us > zonegroup-us.json
```

a. 編集用にファイルを開き、**log_meta** フィールド、**log_data** フィールド、および **sync_from_all** フィールドを **false** に設定します。

例

```
{
    "id": "72f3a886-4c70-420b-bc39-7687f072997d",
    "name": "default",
    "api_name": "",
    "is_master": "true",
    "endpoints": [],
    "hostnames": [],
    "hostnames_s3website": [],
    "master_zone": "a5e44ecd-7aae-4e39-b743-3a709acb60c5",
    "zones": [
        {
            "id": "975558e0-44d8-4866-a435-96d3e71041db",
            "name": "testzone",
            "endpoints": [],
            "log_meta": "false",
            "log_data": "false",
            "bucket_index_max_shards": 11,
            "read_only": "false",
            "tier_type": "",
            "sync_from_all": "false",
            "sync_from": []
        },
        {
            "id": "a5e44ecd-7aae-4e39-b743-3a709acb60c5",
            "name": "default",
            "endpoints": [],
            "log_meta": "false",
            "log_data": "false",
            "bucket_index_max_shards": 11,
            "read_only": "false",
            "tier_type": "",
            "sync_from_all": "false",
            "sync_from": []
        }
    ],
    "placement_targets": [
        {
            "name": "default-placement",
```

```
            "tags": []
        }
    ],
    "default_placement": "default-placement",
    "realm_id": "2d988e7d-917e-46e7-bb18-79350f6a5155"
}
```

5. 更新された JSON ファイルを使用してゾーングループを設定します。

構文

```
radosgw-admin zonegroup set --rgw-zonegroup=ZONE_GROUP_NAME --
infile=JSON_FILE_NAME
```

例

```
[ceph: root@host01 /]# radosgw-admin zonegroup set --rgw-zonegroup=us --
infile=zonegroup-us.json
```

6. 期間を更新します。

例

```
[ceph: root@host01 /]# radosgw-admin period update --commit
```

関連情報

- レルム
- ゾーングループ
- ゾーン
- インストールガイド

5.8. 同じストレージクラスターに複数のレルムの設定

同じストレージクラスターで複数のレルムを設定できます。これは、マルチサイトの高度なユースケースです。同一のストレージクラスター内に複数のレルムを設定することで、ローカルの Ceph Object Gateway クライアントのトラフィックを処理するためのローカルレルムと、セカンダリーサイトに複製されるデータ用のレプリケートされたレルムを使用することができます。

注記

Red Hat では、各レルムに独自の Ceph Object Gateway があることを推奨しています。

前提条件

- ストレージクラスターの 2 つの稼働中の Red Hat Ceph Storage データセンター。
- ストレージクラスター内の各データセンターのアクセスキーおよびシークレットキー。
- すべての Ceph Object Gateway ノードへの root レベルのアクセス。

- 各データセンターには独自のローカルレルムがあります。両方のサイトでレプリケートするレルムを共有する。

手順

1. ストレージクラスターの最初のデータセンターにローカルレルムを 1 つ作成します。

   構文

   ```
   radosgw-admin realm create --rgw-realm=REALM_NAME --default
   ```

   例

   ```
   [ceph: root@host01 /]# radosgw-admin realm create --rgw-realm=ldc1 --default
   ```

2. 最初のデータセンター上に、1 つのローカルマスターゾーングループを作成します。

   構文

   ```
   radosgw-admin zonegroup create --rgw-zonegroup=ZONE_GROUP_NAME --
   endpoints=http://RGW_NODE_NAME:80 --rgw-realm=REALM_NAME --master --default
   ```

   例

   ```
   [ceph: root@host01 /]# radosgw-admin zonegroup create --rgw-zonegroup=ldc1zg --
   endpoints=http://rgw1:80 --rgw-realm=ldc1 --master --default
   ```

3. 最初のデータセンターに 1 つのローカルゾーンを作成します。

   構文

   ```
   radosgw-admin zone create --rgw-zonegroup=ZONE_GROUP_NAME --rgw-
   zone=ZONE_NAME --master --default --endpoints=HTTP_FQDN[,HTTP_FQDN]
   ```

   例

   ```
   [ceph: root@host01 /]# radosgw-admin zone create --rgw-zonegroup=ldc1zg --rgw-
   zone=ldc1z --master --default --endpoints=http://rgw.example.com
   ```

4. 期間をコミットします。

   例

   ```
   [ceph: root@host01 /]# radosgw-admin period update --commit
   ```

5. 適切なレルムおよびゾーンで Ceph Object Gateway デーモンをデプロイするか、設定データベースを更新できます。

   - 配置仕様を使用して Ceph Object Gateway をデプロイします。

     構文

```
ceph orch apply rgw SERVICE_NAME --realm=REALM_NAME --zone=ZONE_NAME --
placement="NUMBER_OF_DAEMONS HOST_NAME_1 HOST_NAME_2"
```

例

```
[ceph: root@host01 /]# ceph orch apply rgw rgw --realm=ldc1 --zone=ldc1z --
placement="1 host01"
```

- Ceph 設定データベースを更新します。

  構文

```
ceph config set client.rgw.SERVICE_NAME rgw_realm REALM_NAME
ceph config set client.rgw.SERVICE_NAME rgw_zonegroup ZONE_GROUP_NAME
ceph config set client.rgw.SERVICE_NAME rgw_zone ZONE_NAME
```

  例

```
[ceph: root@host01 /]# ceph config set client.rgw.rgwsvcid.mons-1.jwgwwp rgw_realm
ldc1
[ceph: root@host01 /]# ceph config set client.rgw.rgwsvcid.mons-1.jwgwwp
rgw_zonegroup ldc1zg
[ceph: root@host01 /]# ceph config set client.rgw.rgwsvcid.mons-1.jwgwwp rgw_zone
ldc1z
```

6. Ceph Object Gateway を再起動します。

注記

NAME 列の **ceph orch ps** コマンドの出力を使用して、SERVICE_TYPE.ID 情報を取得します。

a. ストレージクラスター内の個別のノードで Ceph Object Gateway を再起動するには、以下を実行します。

   構文

```
systemctl restart ceph-CLUSTER_ID@SERVICE_TYPE.ID.service
```

   例

```
[root@host01 ~]# systemctl restart ceph-c4b34c6f-8365-11ba-dc31-
529020a7702d@rgw.realm.zone.host01.gwasto.service
```

b. ストレージクラスター内のすべてのノードで Ceph Object Gateway を再起動するには、以下を実行します。

   構文

```
ceph orch restart SERVICE_TYPE
```

例

```
[ceph: root@host01 /]# ceph orch restart rgw
```

7. ストレージクラスターの 2 番目のデータセンターに、ローカルレルムを 1 つ作成します。

   構文

   ```
   radosgw-admin realm create --rgw-realm=REALM_NAME --default
   ```

   例

   ```
   [ceph: root@host04 /]# radosgw-admin realm create --rgw-realm=ldc2 --default
   ```

8. 2 番目のデータセンターに、1 つのローカルマスターゾーングループを作成します。

   構文

   ```
   radosgw-admin zonegroup create --rgw-zonegroup=ZONE_GROUP_NAME --
   endpoints=http://RGW_NODE_NAME:80 --rgw-realm=REALM_NAME --master --default
   ```

   例

   ```
   [ceph: root@host04 /]# radosgw-admin zonegroup create --rgw-zonegroup=ldc2zg --
   endpoints=http://rgw2:80 --rgw-realm=ldc2 --master --default
   ```

9. 2 番目のデータセンターに 1 つのローカルゾーンを作成します。

   構文

   ```
   radosgw-admin zone create --rgw-zonegroup=ZONE_GROUP_NAME --rgw-
   zone=ZONE_NAME --master --default --endpoints=HTTP_FQDN[, HTTP_FQDN]
   ```

   例

   ```
   [ceph: root@host04 /]# radosgw-admin zone create --rgw-zonegroup=ldc2zg --rgw-
   zone=ldc2z --master --default --endpoints=http://rgw.example.com
   ```

10. 期間をコミットします。

    例

    ```
    [ceph: root@host04 /]# radosgw-admin period update --commit
    ```

11. 適切なレルムおよびゾーンで Ceph Object Gateway デーモンをデプロイするか、設定データベースを更新できます。

    - 配置仕様を使用して Ceph Object Gateway をデプロイします。

      構文

```
ceph orch apply rgw SERVICE_NAME --realm=REALM_NAME --zone=ZONE_NAME --
placement="NUMBER_OF_DAEMONS HOST_NAME_1 HOST_NAME_2"
```

例

```
[ceph: root@host01 /]# ceph orch apply rgw rgw --realm=ldc2 --zone=ldc2z --
placement="1 host01"
```

- Ceph 設定データベースを更新します。

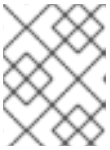
  構文

```
ceph config set client.rgw.SERVICE_NAME rgw_realm REALM_NAME
ceph config set client.rgw.SERVICE_NAME rgw_zonegroup ZONE_GROUP_NAME
ceph config set client.rgw.SERVICE_NAME rgw_zone ZONE_NAME
```

  例

```
[ceph: root@host01 /]# ceph config set client.rgw.rgwsvcid.mons-1.jwgwwp rgw_realm
ldc2
[ceph: root@host01 /]# ceph config set client.rgw.rgwsvcid.mons-1.jwgwwp
rgw_zonegroup ldc2zg
[ceph: root@host01 /]# ceph config set client.rgw.rgwsvcid.mons-1.jwgwwp rgw_zone
ldc2z
```

12. Ceph Object Gateway を再起動します。



注記

NAME 列の **ceph orch ps** コマンドの出力を使用して、SERVICE_TYPE.ID 情報を取得します。

a. ストレージクラスター内の個別のノードで Ceph Object Gateway を再起動するには、以下を実行します。

   構文

```
systemctl restart ceph-CLUSTER_ID@SERVICE_TYPE.ID.service
```

   例

```
[root@host04 ~]# systemctl restart ceph-c4b34c6f-8365-11ba-dc31-
529020a7702d@rgw.realm.zone.host01.gwasto.service
```

b. ストレージクラスター内のすべてのノードで Ceph Object Gateway を再起動するには、以下を実行します。

   構文

```
ceph orch restart SERVICE_TYPE
```

例

```
[ceph: root@host04 /]# ceph orch restart rgw
```

13. ストレージクラスターの最初のデータセンターにレプリケートされたレルムを作成します。

構文

```
radosgw-admin realm create --rgw-realm=REPLICATED_REALM_1 --default
```

例

```
[ceph: root@host01 /]# radosgw-admin realm create --rgw-realm=rdc1 --default
```

--default フラグを使用して、レプリケートされたレルムをプライマリーサイトにデフォルト設定します。

14. 最初のデータセンターのマスターゾーングループを作成します。

構文

```
radosgw-admin zonegroup create --rgw-zonegroup=RGW_ZONE_GROUP --
endpoints=http://_RGW_NODE_NAME:80 --rgw-realm=_RGW_REALM_NAME --master --
default
```

例

```
[ceph: root@host01 /]# radosgw-admin zonegroup create --rgw-zonegroup=rdc1zg --
endpoints=http://rgw1:80 --rgw-realm=rdc1 --master --default
```

15. 最初のデータセンターにマスターゾーンを作成します。

構文

```
radosgw-admin zone create --rgw-zonegroup=RGW_ZONE_GROUP --rgw-
zone=_MASTER_RGW_NODE_NAME --master --default --
endpoints=HTTP_FQDN[,HTTP_FQDN]
```

例

```
[ceph: root@host01 /]# radosgw-admin zone create --rgw-zonegroup=rdc1zg --rgw-
zone=rdc1z --master --default --endpoints=http://rgw.example.com
```

16. 同期ユーザーを作成し、システムユーザーをマルチサイトのマスターゾーンに追加します。

構文

```
radosgw-admin user create --uid="SYNCHRONIZATION_USER" --display-
name="Synchronization User" --system
radosgw-admin zone modify --rgw-zone=RGW_ZONE --access-key=ACCESS_KEY --
secret=SECRET_KEY
```

例

```
radosgw-admin user create --uid="synchronization-user" --display-name="Synchronization
User" --system
[ceph: root@host01 /]# radosgw-admin zone modify --rgw-zone=rdc1zg --access-
key=3QV0D6ZMMCJZMSCXJ2QJ --
secret=VpvQWcsfI9OPzUCpR4kynDLAbqa1OIKqRB6WEnH8
```

17. 期間をコミットします。

例

```
[ceph: root@host01 /]# radosgw-admin period update --commit
```

18. 適切なレルムおよびゾーンで Ceph Object Gateway デーモンをデプロイするか、設定データベースを更新できます。

- 配置仕様を使用して Ceph Object Gateway をデプロイします。

構文

```
ceph orch apply rgw SERVICE_NAME --realm=REALM_NAME --zone=ZONE_NAME --
placement="NUMBER_OF_DAEMONS HOST_NAME_1 HOST_NAME_2"
```

例

```
[ceph: root@host01 /]# ceph orch apply rgw rgw --realm=rdc1 --zone=rdc1z --
placement="1 host01"
```

- Ceph 設定データベースを更新します。

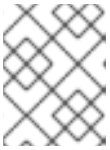
構文

```
ceph config set client.rgw.SERVICE_NAME  rgw_realm REALM_NAME
ceph config set client.rgw.SERVICE_NAME  rgw_zonegroup ZONE_GROUP_NAME
ceph config set client.rgw.SERVICE_NAME  rgw_zone ZONE_NAME
```

例

```
[ceph: root@host01 /]# ceph config set client.rgw.rgwsvcid.mons-1.jwgwwp rgw_realm
rdc1
[ceph: root@host01 /]# ceph config set client.rgw.rgwsvcid.mons-1.jwgwwp
rgw_zonegroup rdc1zg
[ceph: root@host01 /]# ceph config set client.rgw.rgwsvcid.mons-1.jwgwwp rgw_zone
rdc1z
```

19. Ceph Object Gateway を再起動します。



注記

NAME 列の **ceph orch ps** コマンドの出力を使用して、SERVICE_TYPE.ID 情報を取得します。

a. ストレージクラスター内の個別のノードで Ceph Object Gateway を再起動するには、以下を実行します。

構文

```
systemctl restart ceph-CLUSTER_ID@SERVICE_TYPE.ID.service
```

例

```
[root@host01 ~]# systemctl restart ceph-c4b34c6f-8365-11ba-dc31-529020a7702d@rgw.realm.zone.host01.gwasto.service
```

b. ストレージクラスター内のすべてのノードで Ceph Object Gateway を再起動するには、以下を実行します。

構文

```
ceph orch restart SERVICE_TYPE
```

例

```
[ceph: root@host01 /]# ceph orch restart rgw
```

20. 2 番目のデータセンターでレプリケートされたレルムをプルします。

構文

```
radosgw-admin realm pull --url=https://tower-osd1.cephtips.com --access-key=ACCESS_KEY --secret-key=SECRET_KEY
```

例

```
[ceph: root@host01 /]# radosgw-admin realm pull --url=https://tower-osd1.cephtips.com --access-key=3QV0D6ZMMCJZMSCXJ2QJ --secret-key=VpvQWcsfI9OPzUCpR4kynDLAbqa1OIKqRB6WEnH8
```

21. 最初のデータセンターから期間をプルします。

構文

```
radosgw-admin period pull --url=https://tower-osd1.cephtips.com --access-key=ACCESS_KEY --secret-key=SECRET_KEY
```

例

```
[ceph: root@host01 /]# radosgw-admin period pull --url=https://tower-osd1.cephtips.com --access-key=3QV0D6ZMMCJZMSCXJ2QJ --secret-key=VpvQWcsfI9OPzUCpR4kynDLAbqa1OIKqRB6WEnH8
```

22. 2 番目のデータセンターにセカンダリーゾーンを作成します。

構文

```
radosgw-admin zone create --rgw-zone=RGW_ZONE --rgw-
zonegroup=RGW_ZONE_GROUP --endpoints=https://tower-osd4.cephtips.com --
access-key=_ACCESS_KEY --secret-key=SECRET_KEY
```

例

```
[ceph: root@host04 /]# radosgw-admin zone create --rgw-zone=rdc2z --rgw-
zonegroup=rdc1zg --endpoints=https://tower-osd4.cephtips.com --access-
key=3QV0D6ZMMCJZMSCXJ2QJ --secret-
key=VpvQWcsfI9OPzUCpR4kynDLAbqa1OIKqRB6WEnH8
```

23. 期間をコミットします。

例

```
[ceph: root@host04 /]# radosgw-admin period update --commit
```

24. 適切なレルムおよびゾーンで Ceph Object Gateway デーモンをデプロイするか、設定データベースを更新できます。

- 配置仕様を使用して Ceph Object Gateway をデプロイします。

構文

```
ceph orch apply rgw SERVICE_NAME --realm=REALM_NAME --zone=ZONE_NAME --
placement="NUMBER_OF_DAEMONS HOST_NAME_1 HOST_NAME_2"
```

例

```
[ceph: root@host04 /]# ceph orch apply rgw rgw --realm=rdc1 --zone=rdc2z --
placement="1 host04"
```

- Ceph 設定データベースを更新します。

構文

```
ceph config set client.rgw.SERVICE_NAME rgw_realm REALM_NAME
ceph config set client.rgw.SERVICE_NAME rgw_zonegroup ZONE_GROUP_NAME
ceph config set client.rgw.SERVICE_NAME rgw_zone ZONE_NAME
```

例

```
[ceph: root@host04 /]# ceph config set client.rgw.rgwsvcid.mons-1.jwgwwp rgw_realm
rdc1
[ceph: root@host04 /]# ceph config set client.rgw.rgwsvcid.mons-1.jwgwwp
rgw_zonegroup rdc1zg
[ceph: root@host04 /]# ceph config set client.rgw.rgwsvcid.mons-1.jwgwwp rgw_zone
rdc2z
```

25. Ceph Object Gateway を再起動します。

注記

NAME 列の **ceph orch ps** コマンドの出力を使用して、**SERVICE_TYPE.ID** 情報を取得します。

a. ストレージクラスター内の個別のノードで Ceph Object Gateway を再起動するには、以下を実行します。

構文

```
systemctl restart ceph-CLUSTER_ID@SERVICE_TYPE.ID.service
```

例

```
[root@host02 ~]# systemctl restart ceph-c4b34c6f-8365-11ba-dc31-
529020a7702d@rgw.realm.zone.host01.gwasto.service
```

b. ストレージクラスター内のすべてのノードで Ceph Object Gateway を再起動するには、以下を実行します。

構文

```
ceph orch restart SERVICE_TYPE
```

例

```
[ceph: root@host04 /]# ceph orch restart rgw
```

26. 2 番目のデータセンターのエンドポイントに **root** としてログインします。

27. マスターレルムで同期のステータスを確認します。

構文

```
radosgw-admin sync status
```

例

```
[ceph: root@host04 /]# radosgw-admin sync status
          realm 59762f08-470c-46de-b2b1-d92c50986e67 (ldc2)
      zonegroup 7cf8daf8-d279-4d5c-b73e-c7fd2af65197 (ldc2zg)
           zone 034ae8d3-ae0c-4e35-8760-134782cb4196 (ldc2z)
 metadata sync no sync (zone is master)
```

28. 最初のデータセンターのエンドポイントに **root** としてログインします。

29. レプリケーション同期レルムの同期ステータスを確認します。

構文

```
radosgw-admin sync status --rgw-realm RGW_REALM_NAME
```

例

```
[ceph: root@host01 /]# radosgw-admin sync status --rgw-realm rdc1
          realm 73c7b801-3736-4a89-aaf8-e23c96e6e29d (rdc1)
      zonegroup d67cc9c9-690a-4076-89b8-e8127d868398 (rdc1zg)
           zone 67584789-375b-4d61-8f12-d1cf71998b38 (rdc2z)
  metadata sync syncing
                full sync: 0/64 shards
                incremental sync: 64/64 shards
                metadata is caught up with master
      data sync source: 705ff9b0-68d5-4475-9017-452107cec9a0 (rdc1z)
                        syncing
                        full sync: 0/128 shards
                        incremental sync: 128/128 shards
                        data is caught up with source
          realm 73c7b801-3736-4a89-aaf8-e23c96e6e29d (rdc1)
      zonegroup d67cc9c9-690a-4076-89b8-e8127d868398 (rdc1zg)
           zone 67584789-375b-4d61-8f12-d1cf71998b38 (rdc2z)
  metadata sync syncing
                full sync: 0/64 shards
                incremental sync: 64/64 shards
                metadata is caught up with master
      data sync source: 705ff9b0-68d5-4475-9017-452107cec9a0 (rdc1z)
                        syncing
                        full sync: 0/128 shards
                        incremental sync: 128/128 shards
                        data is caught up with source
```

30. ローカルサイトにデータを保存およびアクセスするには、ローカルレルムのユーザーを作成します。

構文

```
radosgw-admin user create --uid="LOCAL_USER" --display-name="Local user" --rgw-realm=_REALM_NAME --rgw-zonegroup=ZONE_GROUP_NAME --rgw-zone=ZONE_NAME
```

例

```
[ceph: root@host04 /]# radosgw-admin user create --uid="local-user" --display-name="Local user" --rgw-realm=ldc1 --rgw-zonegroup=ldc1zg --rgw-zone=ldc1z
```

重要

デフォルトでは、ユーザーはデフォルトのレルムに作成されます。ユーザーがローカルレルム内のデータにアクセスするには、**radosgw-admin** コマンドに **--rgw-realm** 引数が必要です。

5.9. マルチサイト同期ポリシーの使用

ストレージ管理者は、バケットレベルでマルチサイト同期ポリシーを使用して、異なるゾーンのバケット間のデータ移動を制御できます。このようなポリシーは、バケット粒度同期ポリシーと呼ばれます。これまでは、ゾーン内のすべてのバケットが対称的に扱われていました。これは、各ゾーンに指定のバ

ケットのミラーコピーが含まれ、バケットのコピーはすべてのゾーンで同一であることを意味します。同期プロセスでは、バケット同期ソースとバケット同期宛先が同じバケットを参照していることが前提となっています。

重要

バケット同期ポリシーはデータのみに適用され、バケット同期ポリシーの存在に関係なく、マルチサイト内のすべてのゾーンでメタデータが同期されます。バケット同期ポリシーが **allowed** または **forbidden** の場合に作成、変更、または削除されたオブジェクトは、ポリシーが有効になったときに自動的に同期されません。これらのオブジェクトを同期するには、**bucket sync run** コマンドを実行します。

重要

ゾーングループレベルで複数の同期ポリシーが定義されている場合、同時に有効な状態にできるポリシーは1つだけです。必要に応じてポリシーを切り替えることができます。

同期ポリシーは、古いゾーングループ設定 (**sync_from***) よりも優先されます。同期ポリシーは、ゾーングループレベルで設定できます。これが設定されていると、ゾーングループレベルで旧スタイルの設定を置き換えますが、バケットレベルでも設定できます。

重要

バケット同期ポリシーはアーカイブゾーンに適用されます。アーカイブゾーンからの移動は双方向ではなく、すべてのオブジェクトをアクティブゾーンからアーカイブゾーンに移動できます。ただし、アーカイブゾーンは読み取り専用であるため、アーカイブゾーンからアクティブゾーンにオブジェクトを移動することはできません。

ゾーングループのバケット同期ポリシーの例

```
[ceph: root@host01 /]# radosgw-admin sync info --bucket=buck
{
    "sources": [
        {
            "id": "pipe1",
            "source": {
                "zone": "us-east",
                "bucket": "buck:115b12b3-....4409.1"
            },
            "dest": {
                "zone": "us-west",
                "bucket": "buck:115b12b3-....4409.1"
            },
            ...
        }
    ],
    "dests": [
        {
            "id": "pipe1",
            "source": {
                "zone": "us-west",
                "bucket": "buck:115b12b3-....4409.1"
            },
```

```
            "dest": {
                "zone": "us-east",
                "bucket": "buck:115b12b3-....4409.1"
            },
           ...
        },
        {
            "id": "pipe1",
            "source": {
                "zone": "us-west",
                "bucket": "buck:115b12b3-....4409.1"
            },
            "dest": {
                "zone": "us-west-2",
                "bucket": "buck:115b12b3-....4409.1"
            },
           ...
        }
    ],
    ...
}
```

前提条件

- Red Hat Ceph Storage クラスターが実行されている。
- Ceph Monitor ノードへの root レベルのアクセス。
- Ceph Object Gateway ソフトウェアのインストール。

5.9.1. マルチサイト同期ポリシーグループの状態

同期ポリシーでは、データフロー設定のリストを含む複数のグループと、パイプ設定のリストを定義することができます。データフローは、異なるゾーン間のデータの流れを定義します。複数のゾーンが互いにデータを同期する対称的なデータフローを定義でき、データが 1 つのゾーンから別のゾーンに一方向に移動する指向性データフローを定義できます。

パイプは、これらのデータフローを使用することができる実際のバケットと、それに関連付けられるプロパティーを定義します (例: ソースオブジェクト接頭辞)。

同期ポリシーグループには 3 つの状態があります。

- **enabled** - 同期が許可され、有効になっています。
- **allowed** - 同期が許可されています。
- **forbidden** - このグループで定義されている同期は、許可されません。

ゾーンがレプリケートされる場合、同期ポリシーを使用して特定のバケットのレプリケーションを無効にすることができます。ポリシーの競合を解決するには、次のセマンティクスに従う必要があります。

Zonegroup	Bucket	結果
enabled	enabled	enabled

Zonegroup	Bucket	結果
enabled	allowed	enabled
enabled	forbidden	disabled
allowed	enabled	enabled
allowed	allowed	disabled
allowed	forbidden	disabled
forbidden	enabled	disabled
forbidden	allowed	disabled
forbidden	forbidden	disabled

任意の同期ペア (**SOURCE_ZONE**、**SOURCE_BUCKET**)、(**DESTINATION_ZONE**、**DESTINATION_BUCKET**) を反映するように設定されている複数のグループポリシーの場合、次のルールが次の順序で適用されます。

- 1 つの同期ポリシーが **forbidden** の場合でも、同期は **disabled** になります。
- 同期を **allowed** にするには、少なくとも 1 つのポリシーを **enabled** にする必要があります。

このグループの同期状態は、他のグループよりも優先されます。

ポリシーはバケットレベルで定義できます。バケットレベルの同期ポリシーはゾーングループポリシーのデータフローを継承するため、ゾーングループで許可されるもののサブセットのみを定義できます。

ワイルドカードゾーンおよびポリシーのワイルドカードバケットパラメーターは、すべての関連するゾーンまたは関連するバケットをすべて定義します。バケットポリシーのコンテキストでは、現在のバケットインスタンスを意味します。ゾーン全体がミラーリングされている障害復旧設定では、バケットに何も設定する必要がありません。ただし、きめ細かいバケット同期の場合は、ゾーングループレベルで (たとえば、ワイルドカードを使用して) パイプを許可 (**status=allowed**) して同期するように設定することを推奨します。ただし、特定の同期はバケットレベル (**status=enabled**) でのみ有効にします。必要に応じて、バケットレベルのポリシーで、データの移動を特定の関連ゾーンに制限することができます。

重要

ゾーングループポリシーの変更は、ゾーングループマスターゾーンに適用する必要があり、期間更新とコミットが必要です。バケットポリシーへの変更は、ゾーングループマスターゾーンに適用する必要があります。Ceph Object Gateway はこれらの変更を動的に処理します。

S3 バケットレプリケーション API

S3 バケットレプリケーション API も実装され、ユーザーは異なるバケット間でレプリケーションルールを作成できるようになりました。AWS レプリケーション機能により、同じゾーン内のバケットレプ

リケーションが許可されますが、現時点では Ceph Object Gateway は許可しません。ただし、Ceph Object Gateway API には、特定のバケットを同期するゾーンをユーザーが選択できる **Zone** 配列も追加されました。

関連情報

- 詳細については、S3 バケットレプリケーション API を参照してください。

5.9.2. 現在のポリシーの取得

get コマンドを使用して、現在のゾーングループ同期ポリシーまたは特定のバケットポリシーを取得できます。

前提条件

- Red Hat Ceph Storage クラスターが実行されている。
- root または **sudo** アクセス。
- Ceph Object Gateway がインストールされている。

手順

- 現在のゾーングループ同期ポリシーまたはバケットポリシーを取得します。特定のバケットポリシーを取得するには、**--bucket** オプションを使用します。

  構文

  ```
  radosgw-admin sync policy get --bucket=BUCKET_NAME
  ```

  例

  ```
  [ceph: root@host01 /]# radosgw-admin sync policy get --bucket=mybucket
  ```

5.9.3. 同期ポリシーグループの作成

現在のゾーングループまたは特定のバケットの同期ポリシーグループを作成できます。

forbidden から **enabled** に変更になった同期ポリシーグループのバケット詳細レプリケーションの同期ポリシーを作成する場合は、同期プロセスを完了するために手動更新が必要になる場合があります。

たとえば、ポリシーが **forbidden** になっているときにデータが **bucket1** に書き込まれる場合は、ポリシーが **enabled** に変更した後、データがゾーン間で適切に同期されない可能性があります。変更を適切に同期するには、同期ポリシーに対して **bucket sync run** コマンドを実行します。この手順は、ポリシーが **forbidden** ときにバケットが再シャーディングされる場合にも必要です。この場合、ポリシーを有効にした後に、**bucket sync run** コマンドも使用する必要があります。

前提条件

- Red Hat Ceph Storage クラスターが実行されている。
- root または **sudo** アクセス。
- Ceph Object Gateway がインストールされている。

- アーカイブゾーンを作成する場合は、同期ポリシーグループの前にアーカイブゾーンが作成されていることを確認してください。

手順

1. 同期ポリシーグループまたはバケットポリシーを作成します。バケットポリシーを作成するには、**--bucket** オプションを使用します。

   構文

   ```
   radosgw-admin sync group create --bucket=BUCKET_NAME --group-id=GROUP_ID --status=enabled | allowed | forbidden
   ```

   例

   ```
   [ceph: root@host01 /]# radosgw-admin sync group create --group-id=mygroup1 --status=enabled
   ```

2. オプション: バケットの詳細なレプリケーションの同期プロセスを手動で完了します。

   注記

   ポリシーにデータが書き込まれている場合、またはポリシーが **forbidden** になっているときにバケットが再シャーディングされた場合は、バケットの詳細なレプリケーションを備えたアーカイブゾーンの一部として使用する場合は、この手順が必須です。

   構文

   ```
   radosgw-admin bucket sync run
   ```

   例

   ```
   [ceph: root@host01 /]# radosgw-admin bucket sync run
   ```

関連情報

アーカイブゾーンとバケットの詳細なレプリケーションの設定の詳細は、アーカイブゾーンの設定 を参照してください。

5.9.4. 同期ポリシーグループの変更

現在のゾーングループの既存の同期ポリシーグループまたは特定のバケットに対して変更できます。

forbidden から **enabled** に変更になった同期ポリシーグループのバケット詳細レプリケーションの同期ポリシーを変更する場合は、同期プロセスを完了するのに手動更新が必要になる場合があります。

たとえば、ポリシーが **forbidden** になっているときにデータが **bucket1** に書き込まれる場合は、ポリシーが **enabled** に変更した後、データがゾーン間で適切に同期されない可能性があります。変更を適切に同期するには、同期ポリシーに対して **bucket sync run** コマンドを実行します。この手順は、ポリシーが **forbidden** ときにバケットが再シャーディングされる場合にも必要です。この場合、ポリシーを有効にした後に、**bucket sync run** コマンドも使用する必要があります。

前提条件

- Red Hat Ceph Storage クラスターが実行されている。
- root または **sudo** アクセス。
- Ceph Object Gateway がインストールされている。
- アーカイブゾーンを変更する場合は、同期ポリシーグループよりも前にアーカイブゾーンが作成されていることを確認してください。

手順

1. 同期ポリシーグループまたはバケットポリシーを変更します。バケットポリシーを変更するには、**--bucket** オプションを使用します。

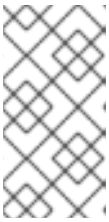
   構文

   ```
   radosgw-admin sync group modify --bucket=BUCKET_NAME --group-id=GROUP_ID --status=enabled | allowed | forbidden
   ```

   例

   ```
   [ceph: root@host01 /]# radosgw-admin sync group modify --group-id=mygroup1 --status=forbidden
   ```

2. オプション: バケットの詳細なレプリケーションの同期プロセスを手動で完了します。

   注記

   ポリシーにデータが書き込まれている場合、またはポリシーが **forbidden** になっているときにバケットが再シャーディングされた場合は、バケットの詳細なレプリケーションを備えたアーカイブゾーンの一部として使用する場合は、この手順が必須です。

   構文

   ```
   radosgw-admin bucket sync run
   ```

   例

   ```
   [ceph: root@host01 /]# radosgw-admin bucket sync run
   ```

関連情報

アーカイブゾーンとバケットの詳細なレプリケーションの設定の詳細は、アーカイブゾーンの設定 を参照してください。

5.9.5. 同期ポリシーグループの取得

group get コマンドを使用して、グループ ID で現在の同期ポリシーグループを表示するか、特定のバケットポリシーを表示できます。

--bucket オプションを指定しない場合は、バケットレベルのグループではなく、ゾーングループレベルで作成されたグループが取得されます。

前提条件

- Red Hat Ceph Storage クラスターが実行されている。
- root または **sudo** アクセス。
- Ceph Object Gateway がインストールされている。

手順

- 現在の同期ポリシーグループまたはバケットポリシーを表示します。特定のバケットポリシーを表示するには、**--bucket** オプションを使用します。

  構文

  ```
  radosgw-admin sync group get --bucket=BUCKET_NAME --group-id=GROUP_ID
  ```

  例

  ```
  [ceph: root@host01 /]# radosgw-admin sync group get --group-id=mygroup
  ```

5.9.6. 同期ポリシーグループの削除

group remove コマンドを使用して、グループ ID で現在の同期ポリシーグループを削除したり、特定のバケットポリシーを削除したりできます。

前提条件

- Red Hat Ceph Storage クラスターが実行されている。
- root または **sudo** アクセス。
- Ceph Object Gateway がインストールされている。

手順

- 現在の同期ポリシーグループまたはバケットポリシーを削除します。特定のバケットポリシーを削除するには、**--bucket** オプションを使用します。

  構文

  ```
  radosgw-admin sync group remove --bucket=BUCKET_NAME --group-id=GROUP_ID
  ```

  例

  ```
  [ceph: root@host01 /]# radosgw-admin sync group remove --group-id=mygroup
  ```

5.9.7. 同期フローの作成

同期ポリシーグループまたは特定のバケットに、異なるタイプのフローを作成できます。

- 指向性同期フロー
- 対称同期フロー

group flow create コマンドは、同期フローを作成します。すでに同期フローがある同期ポリシーグループまたはバケットに対して **group flow create** コマンドを発行すると、コマンドは同期フローの既存の設定を上書きし、指定した設定を適用します。

オプション	説明	必須/オプション
--bucket	同期ポリシーを設定する必要があるバケットの名前。バケットレベルの同期ポリシーでのみ使用されます。	任意
--group-id	同期グループの ID。	必須
--flow-id	フローの ID。	必須
--flow-type	同期ポリシーグループまたは特定のバケットのフロータイプ (指向性または対称性)。	必須
--source-zone	同期元となるソースゾーンを指定します。同期グループにデータを送信するゾーンです。同期グループのフロータイプが指向性である場合は必須です。	任意
--dest-zone	同期先となる宛先ゾーンを指定します。同期グループからデータを受信するゾーンです。同期グループのフロータイプが指向性である場合は必須です。	任意
--zones	同期グループの一部であるゾーン。ゾーンは、送信側ゾーンと受信側ゾーンの両方に言及します。ゾーンは "," で区切って指定します。同期グループのフロータイプが対称の場合は必須です。	任意

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- root または **sudo** アクセス。
- Ceph Object Gateway がインストールされている。

手順

1. 指向性同期フローを作成または更新します。特定のバケットへの指向性同期フローを作成または更新するには、**--bucket** オプションを使用します。

   構文

   ```
   radosgw-admin sync group flow create --bucket=BUCKET_NAME --group-id=GROUP_ID --flow-id=FLOW_ID --flow-type=directional --source-zone=SOURCE_ZONE --dest-zone=DESTINATION_ZONE
   ```

2. 対称同期フローを作成または更新します。対称フロータイプに複数のゾーンを指定するには、**--zones** オプションにコンマ区切りのリストを使用します。

   構文

   ```
   radosgw-admin sync group flow create --bucket=BUCKET_NAME --group-id=GROUP_ID --flow-id=FLOW_ID --flow-type=symmetrical --zones=ZONE_NAME1,ZONE_NAME2
   ```

   zones は、フローに追加する必要があるすべてのゾーンのコンマ区切りのリストです。

5.9.8. 同期フローおよびゾーンの削除

group flow remove コマンドは、同期ポリシーグループまたはバケットから同期フローまたはゾーンを削除します。

指向性フローを使用する同期ポリシーグループまたはバケットの場合は、**group flow remove** コマンドによりフローが削除されます。対称フローを使用した同期ポリシーグループまたはバケットについては、**group flow remove** コマンドを使用してフローから指定されたゾーンを削除したり、フローを削除したりできます。

前提条件

- Red Hat Ceph Storage クラスターが実行されている。
- root または **sudo** アクセス。
- Ceph Object Gateway がインストールされている。

手順

1. 指向性同期フローを削除します。特定のバケットの指向性同期フローを削除するには、**--bucket** オプションを使用します。

   構文

   ```
   radosgw-admin sync group flow remove --bucket=BUCKET_NAME --group-id=GROUP_ID --flow-id=FLOW_ID --flow-type=directional --source-zone=SOURCE_ZONE --dest-zone=DESTINATION_ZONE
   ```

2. 対称同期フローから特定のゾーンを削除します。対称フローから複数のゾーンを削除するには、**--zones** オプションにコンマ区切りのリストを使用します。

   構文

```
radosgw-admin sync group flow remove --bucket=BUCKET_NAME --group-id=GROUP_ID --flow-id=FLOW_ID --flow-type=symmetrical --zones=ZONE_NAME1,ZONE_NAME2
```

3. 対称同期フローを削除します。ゾーングループレベルで同期フローを削除するには、**--bucket** オプションを削除します。

構文

```
radosgw-admin sync group flow remove --group-id=GROUP_ID --flow-id=FLOW_ID --flow-type=symmetrical --zones=ZONE_NAME1,ZONE_NAME2
```

5.9.9. 同期グループパイプの作成または変更

ストレージ管理者は、パイプを定義して、設定したデータフローと、これらのデータフローに関連するプロパティーを使用できるバケットを指定できます。

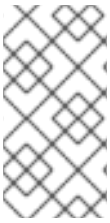
sync group pipe create コマンドを使用すると、パイプを作成できます。パイプは、特定のバケットまたはバケットのグループ間、または特定のゾーンまたはゾーンのグループ間のカスタム同期グループデータフローです。

コマンドは、以下のオプションを使用します。

オプション	説明	必須/オプション
--bucket	同期ポリシーを設定する必要があるバケットの名前。バケットレベルの同期ポリシーでのみ使用されます。	任意
--group-id	同期グループの ID	必須
--pipe-id	パイプの ID	必須
--source-zones	データを同期グループに送信するゾーン。値には一重引用符 (') を使用します。複数のゾーンを分離する場合はコンマで区切ります。データフロールールに一致するすべてのゾーンに、ワイルドカード * を使用します。	必須
--source-bucket	データを同期グループに追加する 1 つまたは複数のバケット。バケット名を指定しない場合は、* (ワイルドカード) がデフォルト値として使用されます。バケットレベルでは、同期グループが作成されたバケットがソースバケットとなり、ゾーングループレベルでは、すべてのバケットがソースバケットとなります。	任意

オプション	説明	必須/オプション
--source-bucket-id	ソースバケットの ID。	任意
--dest-zones	同期データを受け取る 1 つまたは複数のゾーン。値には一重引用符 (') を使用します。複数のゾーンを分離する場合はコンマで区切ります。データフロールールに一致するすべてのゾーンに、ワイルドカード * を使用します。	必須
--dest-bucket	同期データを受け取る 1 つまたは複数のバケット。バケット名を指定しない場合は、* (ワイルドカード) がデフォルト値として使用されます。バケットレベルでは、同期グループが作成されたバケットが宛先バケットとなり、ゾーングループレベルでは、すべてのバケットが宛先バケットとなります。	任意
--dest-bucket-id	宛先バケットの ID。	任意
--prefix	バケットの接頭辞。ワイルドカード * を使用してソースオブジェクトをフィルターします。	任意
--prefix-rm	フィルタリングにバケット接頭辞を使用しないでください。	任意
--tags-add	key=value ペアのコンマ区切りリスト。	任意
--tags-rm	タグの 1 つ以上の key=value ペアを削除します。	任意
--dest-owner	ソースからオブジェクトの宛先所有者。	任意
--storage-class	ソースからのオブジェクトのための宛先ストレージクラス。	任意
--mode	システムモードまたはユーザーモードで それぞれ **system** または **user** を使用します。	任意

オプション	説明	必須/オプション
--uid	ユーザーモードでパーミッションの検証に使用されます。同期操作を発行するユーザー ID を指定します。	任意



注記

ゾーングループレベルで特定のバケットの同期を有効/無効にしたい場合は、ゾーングループレベルの同期ポリシーを有効/無効に設定し、同じバケット名の **--source-bucket** と **--dest-bucket**、または **bucket-id** (つまり **--source-bucket-id** と **--dest-bucket-id**) を使用して、バケットごとにパイプを作成します。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- root または **sudo** アクセス。
- Ceph Object Gateway がインストールされている。

手順

- 同期グループパイプを作成します。**create** コマンドは、関連するオプションのみを指定して同期グループパイプを作成することにより、コマンドを更新するためにも使用されます。

  構文

  ```
  radosgw-admin sync group pipe create --bucket=BUCKET_NAME --group-id=GROUP_ID --pipe-id=PIPE_ID --source-zones='ZONE_NAME','ZONE_NAME2'... --source-bucket=SOURCE_BUCKET --source-bucket-id=SOURCE_BUCKET_ID --dest-zones='ZONE_NAME','ZONE_NAME2'... --dest-bucket=DESTINATION_BUCKET --dest-bucket-id=DESTINATION_BUCKET_ID --prefix=SOURCE_PREFIX --prefix-rm --tags-add=KEY1=VALUE1,KEY2=VALUE2,.. --tags-rm=KEY1=VALUE1,KEY2=VALUE2, ... --dest-owner=OWNER_ID --storage-class=STORAGE_CLASS --mode=USER --uid=USER_ID
  ```

5.9.10. 同期グループパイプの変更または削除

ストレージ管理者は、**sync group pipe modify** コマンドまたは **sync group pipe remove** コマンドを使用して、特定のオプションを削除して同期グループパイプを変更できます。**sync group pipe remove** コマンドを使用して、ゾーン、バケット、または同期グループパイプを完全に削除することもできます。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- root または **sudo** アクセス。
- Ceph Object Gateway がインストールされている。

手順

- 同期グループのパイプオプションを **modify** 引数で変更します。

  構文

  ```
  radosgw-admin sync group pipe modify --bucket=BUCKET_NAME --group-id=GROUP_ID --pipe-id=PIPE_ID --source-zones='ZONE_NAME','ZONE_NAME2'... --source-bucket=SOURCE_BUCKET1 --source-bucket-id=SOURCE_BUCKET_ID --dest-zones='ZONE_NAME','ZONE_NAME2'... --dest-bucket=DESTINATION_BUCKET1 --dest-bucket-id=_DESTINATION_BUCKET-ID
  ```

  注記

  ゾーンは必ず一重引用符 (') で囲んでください。ソースバケットには引用符は必要ありません。

  例

  ```
  [root@host01 ~]# radosgw-admin sync group pipe modify --group-id=zonegroup --pipe-id=pipe --dest-zones='primary','secondary','tertiary' --source-zones='primary','secondary','tertiary' --source-bucket=pri-bkt-1 --dest-bucket=pri-bkt-1
  ```

- 同期グループのパイプオプションを、**remove** 引数を使用して変更します。

  構文

  ```
  radosgw-admin sync group pipe remove --bucket=BUCKET_NAME --group-id=GROUP_ID --pipe-id=PIPE_ID --source-zones='ZONE_NAME','ZONE_NAME2'... --source-bucket=SOURCE_BUCKET, --source-bucket-id=SOURCE_BUCKET_ID --dest-zones='ZONE_NAME','ZONE_NAME2'... --dest-bucket=DESTINATION_BUCKET --dest-bucket-id=DESTINATION_BUCKET-ID
  ```

  例

  ```
  [root@host01 ~]# radosgw-admin sync group pipe remove --group-id=zonegroup --pipe-id=pipe --dest-zones='primary','secondary','tertiary' --source-zones='primary','secondary','tertiary' --source-bucket=pri-bkt-1 --dest-bucket=pri-bkt-1
  ```

- 同期グループパイプを削除します。

  構文

  ```
  radosgw-admin sync group pipe remove --bucket=BUCKET_NAME --group-id=GROUP_ID --pipe-id=PIPE_ID
  ```

  例

  ```
  [root@host01 ~]# radosgw-admin sync group pipe remove -bucket-name=mybuck --group-id=zonegroup --pipe-id=pipe
  ```

5.9.11. 同期操作に関する情報の取得

sync info コマンドを使用すると、同期ポリシーで定義されているように、予想される同期のソースとターゲットに関する情報を取得できます。

バケットの同期ポリシーを作成する場合、そのポリシーはそのバケットから異なるゾーンの別のバケットにデータを移動する方法を定義します。ポリシーを作成すると、バケットを別のバケットと同期するたびにヒントとして使用されるバケット依存関係のリストも作成されます。同期はデータフローが同期を許可しているかどうかによって決まるため、あるバケットが別のバケットを参照していても、実際にはそのバケットと同期していないことがあることに注意してください。

--bucket パラメーターおよび **effective-zone-name** パラメーターはいずれも任意です。オプションを指定せずに **sync info** コマンドを実行すると、Object Gateway はすべてのゾーンの同期ポリシーによって定義されたすべての同期操作を返します。

前提条件

- Red Hat Ceph Storage クラスターが実行されている。
- root または **sudo** アクセス。
- Ceph Object Gateway がインストールされている。
- グループ同期ポリシーが定義されています。

手順

- バケットの同期操作に関する情報を取得します。

  構文

  ```
  radosgw-admin sync info --bucket=BUCKET_NAME --effective-zone-name=ZONE_NAME
  ```

- ゾーングループレベルでの同期操作に関する情報を取得します。

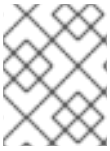
  構文

  ```
  radosgw-admin sync info
  ```

5.10. バケットの詳細な同期ポリシー

次の機能がサポートされるようになりました。

- **グリーンフィールドデプロイメント**: このリリースでは、新しいマルチサイトデプロイメントがサポートされています。バケットの詳細な同期レプリケーションを設定するには、少なくとも新しいゾーングループ/ゾーンを設定する必要があります。
- **ブラウンフィールドデプロイメント**: Ceph Object Gateway のマルチサイトレプリケーション設定を、新しく機能する Ceph Object Gateway バケットの詳細な同期ポリシーレプリケーションに移行またはアップグレードします。



注記

アップグレード中は、ストレージクラスター内のすべてのノードが同じスキーマにあることを確認してください。

- **データフロー - 方向性、対称性**: 単方向レプリケーションと双方向/対称レプリケーションの両方を設定できます。

重要

次の機能は、このリリースではサポートされていません。

- Source **フィルター**
- **ストレージクラス**
- **宛先所有者の翻訳**
- User mode

バケットまたはゾーングループの同期ポリシーが **disabled** 状態から **enabled** 状態に移行すると、次の動作の変化が見られます。

通常シナリオ:

- ゾーングループレベル: 同期ポリシーが **無効になっている** ときに書き込まれたデータは、追加の手順を行わなくても、**有効になる** とすぐにキャッチアップします。
- バケットレベル: 同期ポリシーが **無効な** ときに書き込まれたデータは、ポリシーが **有効な** ときにはキャッチアップしません。この場合、次の 2 つの回避策のいずれかを適用できます。
  - 新しいデータをバケットに書き込むと、古いデータが再同期されます。
  - **Bucket sync run** コマンドを実行すると、すべての古いデータが同期されます。

注記

同期ポリシーからレガシーポリシーに切り替える場合は、最初に **sync init** コマンドを実行し、続いて **radosgw-adminbucket sync run** コマンドを実行してすべてのオブジェクトを同期する必要があります。

リシャードシナリオ:

- ゾーングループレベル: ポリシーが **disabled** なっているときに発生するリシャードは、ポリシーが再度 **enabled** になった後に同期が停止します。この時点では、新しいオブジェクトも同期しません。回避策として **bucket sync run** コマンドを実行します。
- バケットレベル: ポリシーが **disabled** のときにバケットがリシャーディングされると、ポリシーが再度 enabled になった後に同期が停止します。この時点では、新しいオブジェクトも同期しません。回避策として **bucket sync run** コマンドを実行します。

注記

ゾーングループに対してポリシーが **enabled** に設定され、バケットに対してポリシーが **enabled** または **allowed** に設定されている場合、パイプ設定はバケットレベルではなくゾーングループレベルから有効になります。これは既知の問題です (BZ#2240719)。

5.10.1. ゾーングループの双方向ポリシーの設定

ゾーングループ同期ポリシーは、新しい同期ポリシーエンジンを使用して作成されます。ゾーングループ同期ポリシーを変更するには、期間の更新とコミットが必要です。

次の例では、グループポリシーを作成し、あるゾーンから別のゾーンにデータを移動するためのデータフローを定義します。ゾーングループのパイプを設定して、このデータフローを使用できるバケットを定義します。以下の例のシステムには、**us-east** (マスターゾーン)、**us-west**、および **us-west-2** の 3 つのゾーンが含まれています。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- Ceph Object Gateway がインストールされている。

手順

1. ステータスを **allowed** に設定して新しい同期グループを作成します。

   例

   ```
   [ceph: root@host01 /]# radosgw-admin sync group create --group-id=group1 --
   status=allowed
   ```

   注記

   完全に設定されたゾーングループレプリケーションポリシーが作成されるまでは、レプリケーションが開始されないように **--status** を **allowed** に設定することを推奨します。

2. **--flow-type** を **symmetrical** として設定して、新しく作成したグループのフローポリシーを作成し、双方向レプリケーションを有効にします。

   例

   ```
   [ceph: root@host01 /]# radosgw-admin sync group flow create --group-id=group1 \
                        --flow-id=flow-mirror --flow-type=symmetrical \
                        --zones=us-east,us-west
   ```

3. **pipe** という名前の新しいパイプを作成します。

   例

   ```
   [ceph: root@host01 /]# radosgw-admin sync group pipe create --group-id=group1 \
                          --pipe-id=pipe1 --source-zones='*' \
                          --source-bucket='*' --dest-zones='*' \
   ```

```
                --dest-bucket='*'
```

注記

以前のフローポリシーで設定されたすべてのゾーンを含めるにはゾーンに * ワイルドカードを使用し、ゾーン内のすべての既存のバケットを複製するにはバケットに * を使用します。

4. バケット同期ポリシーを設定した後、**--status** を **Enabled** に設定します。

例

```
[ceph: root@host01 /]# radosgw-admin sync group modify --group-id=group1 --
status=enabled
```

5. 新しい期間を更新してコミットします。

例

```
[ceph: root@host01 /]# radosgw-admin period update --commit
```

注記

期間の更新とコミットは、ゾーングループポリシーでは必須です。

6. オプション: 特定のバケットの同期元と同期先を確認します。**us-east** および **us-west** ゾーン内のすべてのバケットは双方向にレプリケートされます。

例

```
[ceph: root@host01 /]# radosgw-admin sync info -bucket buck
{
    "sources": [
        {
            "id": "pipe1",
            "source": {
                "zone": "us-east",
                "bucket": "buck:115b12b3-....4409.1"
            },
            "dest": {
                "zone": "us-west",
                "bucket": "buck:115b12b3-....4409.1"
            },
            ...
        }
    ],
    "dests": [
        {
            "id": "pipe1",
            "source": {
                "zone": "us-west",
                "bucket": "buck:115b12b3-....4409.1"
            },
```

```
            "dest": {
                "zone": "us-east",
                "bucket": "buck:115b12b3-....4409.1"
            },
            ...
        }
    ],
    ...
}
```

上記の出力の **id** フィールドは、そのエントリーを生成したパイプルールを反映しています。以下の例に示すように、1 つのルールで複数の同期エントリーを生成できます。

5.10.2. ゾーングループの方向性ポリシーの設定

同期ポリシーエンジンを使用して、ゾーングループのポリシーを一方向に設定します。

次の例では、グループポリシーを作成し、あるゾーンから別のゾーンにデータを移動するためのデータフローを設定します。ゾーングループのパイプを設定して、このデータフローを使用できるバケットを定義します。次の例のシステムには、**us-east** (プライマリーゾーン)、**us-west** (セカンダリーゾーン)、および **us-west-2** (バックアップゾーン) の 3 つのゾーンが含まれています。ここで、**us-west-2** は **us-west** のレプリカですが、データはそこから複製されません。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- Ceph Object Gateway がインストールされている。

手順

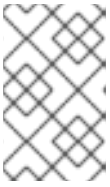
1. プライマリーゾーンで、ステータスを **allowed** に設定して新しい同期グループを作成します。

   構文

   ```
   radosgw-admin sync group create --group-id=GROUP_ID --status=allowed
   ```

   例

   ```
   [ceph: root@host01 /]# radosgw-admin sync group create --group-id=group1 --
   status=allowed
   ```

   注記

   完全に設定されたゾーングループレプリケーションポリシーが作成されるまでは、レプリケーションが開始されないように **--status** を **allowed** に設定することを推奨します。

2. フローを作成します。

   構文

```
radosgw-admin sync group flow create --group-id=GROUP_ID --flow-id=FLOW_ID --flow-type=directional --source-zone=SOURCE_ZONE_NAME --dest-zone=DESTINATION_ZONE_NAME
```

例

```
[ceph: root@host01 /]# radosgw-admin sync group flow create --group-id=group1 --flow-id=us-west-backup --flow-type=directional --source-zone=us-west --dest-zone=us-west-2
```

3. パイプを作成します。

構文

```
radosgw-admin sync group pipe create --group-id=GROUP_ID --pipe-id=PIPE_ID  --source-zones='SOURCE_ZONE_NAME' --dest-zones='DESTINATION_ZONE_NAME'
```

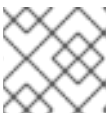
例

```
[ceph: root@host01 /]# radosgw-admin sync group pipe create --group-id=group1 --pipe-id=pipe1 --source-zones='us-west' --dest-zones='us-west-2'
```

4. 新しい期間を更新してコミットします。

例

```
[ceph: root@host01 /]# radosgw-admin period update --commit
```



注記

期間の更新とコミットは、ゾーングループポリシーでは必須です。

5. 両方のサイトの同期情報を使用して、ゾーングループの送信元と宛先を確認します。

構文

```
radosgw-admin sync info
```

5.10.3. バケットの方向性ポリシーの設定

同期ポリシーエンジンを使用してバケットのポリシーを一方向に設定します。

次の例では、グループポリシーを作成し、あるゾーンから別のゾーンにデータを移動するためのデータフローを設定します。ゾーングループのパイプを設定して、このデータフローを使用できるバケットを定義します。次の例のシステムには、**us-east** (プライマリーゾーン)、**us-west** (セカンダリーゾーン)、および **us-west-2** (バックアップゾーン) の 3 つのゾーンが含まれています。ここで、**us-west-2** は **us-west** のレプリカですが、データはそこから複製されません。

ゾーングループとバケットの方向ポリシーの設定の違いは、**--bucket** オプションを指定する必要があることです。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- Ceph Object Gateway がインストールされている。

手順

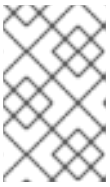
1. プライマリーゾーンで、ステータスを **allowed** に設定して新しい同期グループを作成します。

   構文

   ```
   radosgw-admin sync group create --group-id=GROUP_ID --status=allowed --
   bucket=BUCKET_NAME
   ```

   例

   ```
   [ceph: root@host01 /]# radosgw-admin sync group create --group-id=group1 --
   status=allowed --bucket=buck
   ```

   注記

   完全に設定されたゾーングループレプリケーションポリシーが作成されるまでは、レプリケーションが開始されないように **--status** を **allowed** に設定することを推奨します。

2. フローを作成します。

   構文

   ```
   radosgw-admin sync group flow create --bucket-name=BUCKET_NAME --group-
   id=GROUP_ID --flow-id=FLOW_ID --flow-type=directional --source-
   zone=SOURCE_ZONE_NAME --dest-zone=DESTINATION_ZONE_NAME
   ```

   例

   ```
   [ceph: root@host01 /]# radosgw-admin sync group flow create --bucket-name=buck --group-
   id=group1 --flow-id=us-west-backup --flow-type=directional --source-zone=us-west --dest-
   zone=us-west-2
   ```

3. パイプを作成します。

   構文

   ```
   radosgw-admin sync group pipe create --group-id=GROUP_ID --bucket-
   name=BUCKET_NAME --pipe-id=PIPE_ID  --source-zones='SOURCE_ZONE_NAME' --
   dest-zones='DESTINATION_ZONE_NAME'
   ```

   例

   ```
   [ceph: root@host01 /]# radosgw-admin sync group pipe create --group-id=group1 --bucket-
   name=buck --pipe-id=pipe1 --source-zones='us-west' --dest-zones='us-west-2'
   ```

4. 両方のサイトの同期情報を使用して、ゾーングループの送信元と宛先を確認します。

   構文

   ```
   radosgw-admin sync info --bucket-name=BUCKET_NAME
   ```

5.10.4. バケットの双方向ポリシーの設定

バケットレベルのポリシーのデータフローは、ゾーングループポリシーから継承されます。バケットレベルのポリシーのフローとパイプは、ゾーングループポリシーで定義されたフローのサブセットにすぎないため、バケットレベルのポリシーのデータフローとパイプを変更する必要はありません。

注記

- バケットレベルのポリシーでは、ゾーングループポリシーで **禁止されている** パイプを除き、有効になっていないパイプを **有効にする** ことができます。
- バケットレベルのポリシーでは期間の更新は必要ありません。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- Ceph Object Gateway がインストールされている。
- 同期フローが作成されます。

手順

1. ゾーングループポリシー **--status** を **allowed** に設定して、バケットごとのレプリケーションを許可します。

   例

   ```
   [ceph: root@host01 /]# radosgw-admin sync group modify --group-id=group1 --
   status=allowed
   ```

2. ゾーングループポリシーを変更した後、期間を更新します。

   例

   ```
   [ceph: root@host01 /]# radosgw-admin period update --commit
   ```

3. 同期するバケットの同期グループを作成し、**--status** を **Enabled** に設定します。

   例

   ```
   [ceph: root@host01 /]# radosgw-admin sync group create --bucket=buck \
                         --group-id=buck-default --status=enabled
   ```

4. 前の手順で作成したグループのパイプを作成します。フローは、データフローが対称であるゾーングループレベルのポリシーから継承されます。

例

```
[ceph: root@host01 /]# radosgw-admin sync group pipe create --bucket=buck \
                      --group-id=buck-default --pipe-id=pipe1 \
                      --source-zones='*' --dest-zones='*'
```

注記

ワイルドカード * を使用して、バケットレプリケーションのソースゾーンと宛先ゾーンを指定します。

5. オプション: 同期ポリシーで定義されている、想定されるバケット同期ソースおよびターゲットに関する情報を取得するには、**--bucket** フラグを指定して **radosgw-admin Bucket sync info** コマンドを実行します。

例

```
[ceph: root@host01 /]# radosgw-admin bucket sync info --bucket buck
realm 33157555-f387-44fc-b4b4-3f9c0b32cd66 (india)
zonegroup 594f1f63-de6f-4e1e-90b6-105114d7ad55 (shared)
zone ffaa5ba4-c1bd-4c17-b176-2fe34004b4c5 (primary)
bucket :buck[ffaa5ba4-c1bd-4c17-b176-2fe34004b4c5.16191.1]

source zone e0e75beb-4e28-45ff-8d48-9710de06dcd0
bucket :buck[ffaa5ba4-c1bd-4c17-b176-2fe34004b4c5.16191.1]
```

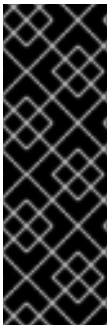
6. オプション: 同期ポリシーで定義されている、想定される同期ソースおよびターゲットに関する情報を取得するには、**--bucket** フラグを指定して **radosgw-admin sync info** コマンドを実行します。

例

```
[ceph: root@host01 /]# radosgw-admin sync info --bucket buck
{
    "id": "pipe1",
    "source": {
        "zone": "secondary",
        "bucket": "buck:ffaa5ba4-c1bd-4c17-b176-2fe34004b4c5.16191.1"
    },
    "dest": {
        "zone": "primary",
        "bucket": "buck:ffaa5ba4-c1bd-4c17-b176-2fe34004b4c5.16191.1"
    },
    "params": {
        "source": {
            "filter": {
                "tags": []
            }
        },
        "dest": {},
        "priority": 0,
        "mode": "system",
        "user": ""
    }
```

```
},
    {
    "id": "pipe1",
    "source": {
        "zone": "primary",
        "bucket": "buck:ffaa5ba4-c1bd-4c17-b176-2fe34004b4c5.16191.1"
    },
    "dest": {
        "zone": "secondary",
        "bucket": "buck:ffaa5ba4-c1bd-4c17-b176-2fe34004b4c5.16191.1"
    },
    "params": {
        "source": {
            "filter": {
                "tags": []
            }
        },
        "dest": {},
        "priority": 0,
        "mode": "system",
        "user": ""
    }
}
```

5.10.5. バケット間の同期 (テクノロジープレビュー)



重要

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

ゾーンをまたいでソースバケットと宛先バケットの間でデータを同期することはできますが、同一ゾーン内で同期することはできません。内部では、データは引き続き宛先ゾーンでソースからプルされることに注意してください。

ワイルドカードバケット名は、現在のバケットがバケット同期ポリシーのコンテキスト内にあることを意味します。

バケット間の同期には 2 つのタイプがあります。

1. バケットからの同期 - ソースバケットを指定する必要があります。
2. バケットへの同期 - 宛先バケットを指定する必要があります。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- Ceph Object Gateway がインストールされている。

別のバケットからの同期

1. 同期グループを作成して、別のゾーンのバケットからデータをプルします。

   構文

   ```
   radosgw-admin sync group create  --bucket=BUCKET_NAME --group-id=GROUP_ID --
   status=enabled
   ```

   例

   ```
   [ceph: root@host01 /]# radosgw-admin sync group create --bucket=buck4 --group-id=buck4-
   default --status=enabled
   ```

2. データをプルします。

   構文

   ```
   radosgw-admin sync group pipe create --bucket-name=BUCKET_NAME --group-
   id=GROUP_ID --pipe-id=PIPE_ID --source-zones=SOURCE_ZONE_NAME --source-
   bucket=SOURCE_BUCKET_NAME --dest-zones=DESTINATION_ZONE_NAME
   ```

   例

   ```
   [ceph: root@host01 /]# radosgw-admin sync group pipe create --bucket=buck4 \
                         --group-id=buck4-default --pipe-id=pipe1 \
                         --source-zones='*' --source-bucket=buck5 \
                         --dest-zones='*'
   ```

   この例では、ソースバケットが **buck5** であることがわかります。

3. オプション: 特定のゾーンのバケットから同期します。

   例

   ```
   [ceph: root@host01 /]# radosgw-admin sync group pipe modify --bucket=buck4 \
                         --group-id=buck4-default --pipe-id=pipe1 \
                         --source-zones=us-west --source-bucket=buck5 \
                         --dest-zones='*'
   ```

4. 同期のステータスを確認します。

   構文

   ```
   radosgw-admin sync info --bucket-name=BUCKET_NAME
   ```

   例

   ```
   [ceph: root@host01 /]# radosgw-admin sync info --bucket=buck4
   {
       "sources": [],
       "dests": [],
   ```

```
    "hints": {
        "sources": [],
        "dests": [
            "buck4:115b12b3-....14433.2"
        ]
    },
    "resolved-hints-1": {
        "sources": [],
        "dests": [
            {
                "id": "pipe1",
                "source": {
                    "zone": "us-west",
                    "bucket": "buck5"
                },
                "dest": {
                    "zone": "us-east",
                    "bucket": "buck4:115b12b3-....14433.2"
                },
                ...
            },
            {
                "id": "pipe1",
                "source": {
                    "zone": "us-west",
                    "bucket": "buck5"
                },
                "dest": {
                    "zone": "us-west-2",
                    "bucket": "buck4:115b12b3-....14433.2"
                },
                ...
            }
        ]
    },
    "resolved-hints": {
        "sources": [],
        "dests": []
    }
```

resolved-hints が存在することに注意してください。これは、バケット **buck5** が、その独自のポリシーからではなく、同バケットから間接的に同期する **buck4** を検出したことを意味します。**buck5** のポリシー自体は空です。

別のバケットへの同期

1. 同期グループを作成します。

構文

```
radosgw-admin sync group create  --bucket=BUCKET_NAME --group-id=GROUP_ID --
status=enabled
```

例

```
[ceph: root@host01 /]# radosgw-admin sync group create --bucket=buck6 \
                      --group-id=buck6-default --status=enabled
```

2. データをプッシュします。

構文

```
radosgw-admin sync group pipe create --bucket-name=BUCKET_NAME --group-
id=GROUP_ID --pipe-id=PIPE_ID --source-zones=SOURCE_ZONE_NAME --dest-
zones=DESTINATION_ZONE_NAME --dest-bucket=DESTINATION_BUCKET_NAME
```

例

```
[ceph: root@host01 /]# radosgw-admin sync group pipe create --bucket=buck6 \
                      --group-id=buck6-default --pipe-id=pipe1 \
                      --source-zones='*' --dest-zones='*' --dest-bucket=buck5
```

この例では、宛先バケットが **buck5** であることがわかります。

3. オプション: 特定のゾーンのバケットに同期します。

例

```
[ceph: root@host01 /]# radosgw-admin sync group pipe modify  --bucket=buck6 \
                      --group-id=buck6-default --pipe-id=pipe1 \
                      --source-zones='*' --dest-zones='us-west' --dest-bucket=buck5
```

4. 同期のステータスを確認します。

構文

```
radosgw-admin sync info --bucket-name=BUCKET_NAME
```

例

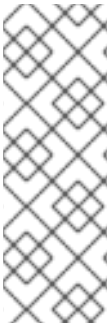
```
[ceph: root@host01 /]#  radosgw-admin sync info --bucket buck5
{
  "sources": [],
  "dests": [
    {
      "id": "pipe1",
      "source": {
        "zone": "us-west",
        "bucket": "buck6:c7887c5b-f6ff-4d5f-9736-aa5cdb4a15e8.20493.4"
      },
      "dest": {
        "zone": "us-east",
        "bucket": "buck5"
      },
      "params": {
        "source": {
          "filter": {
            "tags": []
```

```
            }
          },
          "dest": {},
          "priority": 0,
          "mode": "system",
          "user": "s3cmd"
        }
      },
    ],
    "hints": {
      "sources": [],
      "dests": [
        "buck5"
      ]
    },
    "resolved-hints-1": {
      "sources": [],
      "dests": []
    },
    "resolved-hints": {
      "sources": [],
      "dests": []
    }
  }
```

5.10.6. オブジェクトのフィルタリング

接頭辞とタグを使用してバケット内のオブジェクトをフィルターします。ゾーングループレベルのポリシーでもオブジェクトフィルターを設定できます。**--bucket** オプションを使用すると、バケットレベルで設定されます。

次の例では、あるゾーンの **buck1** バケットからの同期が、接頭辞 **foo/** で始まるオブジェクトを持つ別のゾーンの **buck1** バケットに同期されます。同様に、**color=blue** などのタグを持つオブジェクトをフィルタリングできます。接頭辞とタグは組み合わせることができますが、同期するにはオブジェクトに両方が必要です。priority パラメーターを渡すこともでき、一致する複数の異なるルールが存在するかどうかを判断するために使用されます。この設定により、使用するルールが決まります。



注記

1. 同期ポリシーのタグに複数のタグがある場合、オブジェクトの同期中に、少なくとも 1 つのタグ (キーと値のペア) に一致するオブジェクトが同期されます。オブジェクトはすべてのタグセットと一致しない可能性があります。
2. 接頭辞とタグが設定されている場合、オブジェクトを別のゾーンに同期するには、オブジェクトに接頭辞があり、いずれかのタグが一致する必要があります。そうして初めて、相互に同期します。

前提条件

- 少なくとも 2 つの実行中の Red Hat Ceph Storage クラスター
- Ceph Object Gateway がインストールされている。
- バケットが作成されます。

手順

1. 新しい同期グループを作成します。バケットレベルで作成する場合は、**--bucket** オプションを使用します。

   構文

   ```
   radosgw-admin sync group create --bucket=BUCKET_NAME --group-id=GROUP_ID --status=enabled
   ```

   例

   ```
   [ceph: root@host01 /]# radosgw-admin sync group create --bucket=buck1 --group-id=buck8-default --status=enabled
   ```

2. オブジェクトがタグと一致するバケット間で同期します。フローは、データフローが対称であるゾーングループレベルのポリシーから継承されます。

   構文

   ```
   radosgw-admin sync group pipe create --bucket=BUCKET_NAME --group-id=GROUP_ID --pipe-id=PIPE_ID --tags-add=KEY1=VALUE1,KEY2=VALUE2 --source-zones='ZONE_NAME1','ZONE_NAME2' --dest-zones='ZONE_NAME1','ZONE_NAME2'
   ```

   例

   ```
   [ceph: root@host01 /]# radosgw-admin sync group pipe create --bucket=buck1 \
                        --group-id=buck1-default --pipe-id=pipe-tags \
                        --tags-add=color=blue,color=red --source-zones='*' \
                        --dest-zones='*'
   ```

3. オブジェクトが接頭辞と一致するバケット間で同期します。フローは、データフローが対称であるゾーングループレベルのポリシーから継承されます。

   構文

   ```
   radosgw-admin sync group pipe create --bucket=BUCKET_NAME --group-id=GROUP_ID --pipe-id=PIPE_ID --prefix=PREFIX --source-zones='ZONE_NAME1','ZONE_NAME2' --dest-zones='ZONE_NAME1','ZONE_NAME2'
   ```

   例

   ```
   [ceph: root@host01 /]# radosgw-admin sync group pipe create --bucket=buck1 \
                        --group-id=buck1-default --pipe-id=pipe-prefix \
                        --prefix=foo/ --source-zones='*' --dest-zones='*' \
   ```

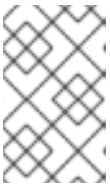
4. 更新された同期を確認します。

   構文

   ```
   radosgw-admin sync info --bucket=BUCKET_NAME
   ```

例

```
[ceph: root@host01 /]# radosgw-admin sync info --bucket=buck1
```



注記

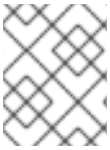
この例では、2 つの異なる宛先のみが反映され、ソースは反映されず、それぞれ 1 つが設定用に使用されます。同期プロセスが発生すると、同期する各オブジェクトに関連するルールが選択されます。

5.10.7. バケット間でのポリシーの無効化

forbidden または **allowed** 状態の同期情報とともにバケット間のポリシーを無効にすることができます。

ゾーングループおよびバケットレベルの同期ポリシーに使用できるさまざまな組み合わせについては、**マルチサイト同期ポリシーグループの状態** を参照してください。

場合によって、2 つのバケット間のレプリケーションを中断するには、バケットのグループポリシーを **forbidden** に設定します。設定した同期がどのバケットでも行われない場合は、ゾーングループレベルでポリシーを無効にすることもできます。



注記

radosgw-admin sync group create コマンドを使用して、**allowed** または **forbidden** 状態で無効状態の同期ポリシーを作成することもできます。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- Ceph Object Gateway がインストールされている。

手順

1. **sync group modify** コマンドを実行して、ステータスを **allowed** から **forbidden** に変更します。

   例

```
[ceph: root@host01 /]# radosgw-admin sync group modify --group-id  buck-default --status
forbidden --bucket buck
{
    "groups": [
        {
            "id": "buck-default",
            "data_flow": {},
            "pipes": [
                {
                    "id": "pipe1",
                    "source": {
                        "bucket": "*",
                        "zones": [
                            "*"
```

```
                ]
            },
            "dest": {
                "bucket": "*",
                "zones": [
                    "*"
                ]
            },
            "params": {
                "source": {
                    "filter": {
                        "tags": []
                    }
                },
                "dest": {},
                "priority": 0,
                "mode": "system",
            }
        }
    ],
    "status": "forbidden"
  }
 ]
}
```

この例では、バケット **buck** のレプリケーションがゾーン **us-east** と **us-west** の間で中断されます。

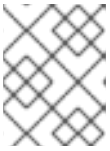
注記

これはバケット同期ポリシーであるため、その期間の更新とコミットは必要ありません。

2. オプション: **sync info command** コマンドを実行して、バケット **buck** の同期のステータスを確認します。

例

```
[ceph: root@host01 /]# radosgw-admin sync info --bucket buck
{
    "sources": [],
    "dests": [],
    "hints": {
        "sources": [],
        "dests": []
    },
    "resolved-hints-1": {
        "sources": [],
        "dests": []
    },
    "resolved-hints": {
        "sources": [],
        "dests": []
    }
}
```



注記

レプリケーションが中断されるため、ソースおよび宛先のターゲットは存在しません。

5.11. マルチサイトの CEPH OBJECT GATEWAY コマンドラインの使用

ストレージ管理者は、マルチサイト環境での Ceph Object Gateway の使用方法を理解することができます。マルチサイト環境で、レルム、ゾーングループ、およびゾーンをより適切に管理する方法を説明します。

前提条件

- 実行中の Red Hat Ceph Storage。
- Ceph Object Gateway ソフトウェアのデプロイメント。
- Ceph Object Gateway ノードまたはコンテナーへのアクセス。

5.11.1. レルム

レルムは、1つ以上のゾーンが含まれる1つ以上のゾーングループと、バケットが含まれるゾーンで設定されるグローバル固有の名前空間を表します。この名前空間にはオブジェクトが含まれます。レルムにより、Ceph Object Gateway は同じハードウェアで複数の名前空間および設定をサポートできるようになります。

レルムには期間の概念が含まれます。それぞれの期間は、ゾーングループとゾーン設定の状態を時間で表しています。ゾーングループまたはゾーンに変更を加えるたびに、期間を更新してコミットします。

Red Hat は新規クラスターのレルムを作成することを推奨します。

5.11.1.1. レルムの作成

レルムを作成するには、**realm create** コマンドを発行してレルム名を指定します。レルムがデフォルトの場合は、**--default** を指定します。

構文

```
radosgw-admin realm create --rgw-realm=REALM_NAME [--default]
```

例

```
[ceph: root@host01 /]# radosgw-admin realm create --rgw-realm=test_realm --default
```

--default を指定すると、**--rgw-realm** とレルム名が明示的に指定されていない限り、各 **radosgw-admin** 呼び出しでレルムが暗黙的に呼び出されます。

5.11.1.2. レルムのデフォルトの設定

レルムリストにある1つのレルムはデフォルトのレルムである必要があります。デフォルトレルムは1つのみです。レルムが1つだけあり、そのレルムが作成時にデフォルトレルムとして指定されていない場合は、デフォルトのレルムにします。または、デフォルトであるレルムを変更するには、以下のコマンドを実行します。

-

```
[ceph: root@host01 /]# radosgw-admin realm default --rgw-realm=test_realm
```

注記

レルムがデフォルトの場合、コマンドラインでは **--rgw-realm=REALM_NAME** を引数と想定します。

5.11.1.3. レルムの削除

レルムを削除するには、**realm delete** コマンドを実行して、レルム名を指定します。

構文

```
radosgw-admin realm delete --rgw-realm=REALM_NAME
```

例

```
[ceph: root@host01 /]# radosgw-admin realm delete --rgw-realm=test_realm
```

5.11.1.4. レルムの取得

レルムを取得するには、**realm get** コマンドを実行してレルム名を指定します。

構文

```
radosgw-admin realm get --rgw-realm=REALM_NAME
```

例

```
[ceph: root@host01 /]# radosgw-admin realm get --rgw-realm=test_realm >filename.json
```

CLI は、レルムプロパティーを使用して JSON オブジェクトを echo します。

```
{
    "id": "0a68d52e-a19c-4e8e-b012-a8f831cb3ebc",
    "name": "test_realm",
    "current_period": "b0c5bbef-4337-4edd-8184-5aeab2ec413b",
    "epoch": 1
}
```

> と出力ファイル名を使用して、JSON オブジェクトをファイルに出力します。

5.11.1.5. レルムの設定

レルムを設定するには、**realm set** コマンドを実行し、レルム名を指定し、**--infile=** を入力ファイル名で指定します。

構文

```
radosgw-admin realm set --rgw-realm=REALM_NAME --infile=IN_FILENAME
```

例

```
[ceph: root@host01 /]# radosgw-admin realm set --rgw-realm=test_realm --infile=filename.json
```

5.11.1.6. レルムのリスト表示

レルムをリスト表示するには、**realm list** コマンドを実行します。

例

```
[ceph: root@host01 /]# radosgw-admin realm list
```

5.11.1.7. レルム期間のリスト表示

レルムの期間をリスト表示するには、**realm list-periods** コマンドを実行します。

例

```
[ceph: root@host01 /]# radosgw-admin realm list-periods
```

5.11.1.8. レルムのプル

マスターゾーングループとマスターゾーンを含むノードからセカンダリーゾーングループまたはゾーンを含むノードにレルムをプルするには、レルム設定を受け取るノードで **realm pull** コマンドを実行します。

構文

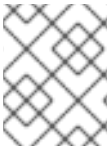
```
radosgw-admin realm pull --url=URL_TO_MASTER_ZONE_GATEWAY--access-
key=ACCESS_KEY --secret=SECRET_KEY
```

5.11.1.9. レルムの名前変更

レルムは期間の一部ではありません。そのため、レルムの名前変更はローカルでのみ適用され、**realm pull** でプルされません。複数のゾーンを持つレルムの名前を変更する場合は、**各ゾーンでこのコマンドを実行します**。レルムの名前を変更するには、以下のコマンドを実行します。

構文

```
radosgw-admin realm rename --rgw-realm=REALM_NAME --realm-new-
name=NEW_REALM_NAME
```



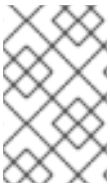
注記

realm set を使用して **name** パラメーターを変更しないでください。これにより、内部名のみが変更されます。**--rgw-realm** を指定すると、古いレルム名が使用されます。

5.11.2. ゾーングループ

Ceph Object Gateway は、ゾーングループの概念を使用したマルチサイトデプロイメントおよびグローバル名前空間をサポートします。以前はリージョンと呼ばれていたゾーングループは、1 つ以上のゾーン内の 1 つ以上の Ceph Object Gateway インスタンスの地理的な場所を定義します。

ゾーングループの設定は、設定のすべてが Ceph 設定ファイルになるわけではないため、一般的な設定手順とは異なります。ゾーングループのリストを表示し、ゾーングループ設定を取得し、ゾーングループ設定を設定できます。



注記

期間を更新するステップはクラスター全体に変更を伝播するため、**radosgw-admin zonegroup** 操作はレルム内の任意のノードで実行できます。ただし、**radosgw-admin zone** 操作は、ゾーン内のホストで実行する **必要があります**。

5.11.2.1. ゾーングループの作成

ゾーングループの作成は、ゾーングループ名の指定から始まります。ゾーンの作成では、**--rgw-realm=REALM_NAME** が指定されていない限り、デフォルトのレルムで実行されていることを前提としています。ゾーングループがデフォルトのゾーングループの場合は、**--default** フラグを指定します。ゾーングループがマスターゾーングループの場合は、**--master** フラグを指定します。

構文

```
radosgw-admin zonegroup create --rgw-zonegroup=ZONE_GROUP_NAME [--rgw-realm=REALM_NAME] [--master] [--default]
```

注記

zonegroup modify --rgw-zonegroup=ZONE_GROUP_NAME を使用して、既存のゾーングループの設定を変更します。

5.11.2.2. ゾーングループをデフォルトにする

ゾーングループリスト内の 1 つのゾーングループは、デフォルトのゾーングループである必要があります。デフォルトのゾーングループは 1 つのみです。ゾーングループが 1 つだけあり、そのゾーンは作成時にデフォルトのゾーングループとして指定されていない場合は、デフォルトのゾーングループにします。または、デフォルトであるゾーングループを変更するには、以下のコマンドを実行します。

例

```
[ceph: root@host01 /]# radosgw-admin zonegroup default --rgw-zonegroup=us
```

注記

ゾーングループがデフォルトの場合、コマンドラインは **--rgw-zonegroup=ZONE_GROUP_NAME** を引数として想定します。

次に、期間を更新します。

```
[ceph: root@host01 /]# radosgw-admin period update --commit
```

5.11.2.3. ゾーングループへのゾーンの追加

ゾーングループにゾーンを追加するには、ゾーンに追加するホストでこのコマンドを実行する**必要があります**。ゾーングループにゾーンを追加するには、次のコマンドを実行します。

構文

```
radosgw-admin zonegroup add --rgw-zonegroup=ZONE_GROUP_NAME --rgw-zone=ZONE_NAME
```

次に、期間を更新します。

例

```
[ceph: root@host01 /]# radosgw-admin period update --commit
```

5.11.2.4. ゾーングループからのゾーンの削除

ゾーングループからゾーンを削除するには、次のコマンドを実行します。

構文

```
radosgw-admin zonegroup remove --rgw-zonegroup=ZONE_GROUP_NAME --rgw-zone=ZONE_NAME
```

次に、期間を更新します。

例

```
[ceph: root@host01 /]# radosgw-admin period update --commit
```

5.11.2.5. ゾーングループの名前変更

ゾーングループの名前を変更するには、次のコマンドを実行します。

構文

```
radosgw-admin zonegroup rename --rgw-zonegroup=ZONE_GROUP_NAME --zonegroup-new-name=NEW_ZONE_GROUP_NAME
```

次に、期間を更新します。

例

```
[ceph: root@host01 /]# radosgw-admin period update --commit
```

5.11.2.6. ゾーングループの削除

ゾーングループを削除するには、次のコマンドを実行します。

構文

```
radosgw-admin zonegroup delete --rgw-zonegroup=ZONE_GROUP_NAME
```

次に、期間を更新します。

例

```
[ceph: root@host01 /]# radosgw-admin period update --commit
```

5.11.2.7. ゾーングループのリスト表示

Ceph クラスターには、ゾーングループのリストが含まれます。ゾーングループをリスト表示するには、以下のコマンドを実行します。

```
[ceph: root@host01 /]# radosgw-admin zonegroup list
```

radosgw-admin は、JSON 形式のゾーングループのリストを返します。

```
{
    "default_info": "90b28698-e7c3-462c-a42d-4aa780d24eda",
    "zonegroups": [
        "us"
    ]
}
```

5.11.2.8. ゾーングループの取得

ゾーングループの設定を表示するには、次のコマンドを実行します。

構文

```
radosgw-admin zonegroup get [--rgw-zonegroup=ZONE_GROUP_NAME]
```

ゾーングループの設定は以下のようになります。

```
{
    "id": "90b28698-e7c3-462c-a42d-4aa780d24eda",
    "name": "us",
    "api_name": "us",
    "is_master": "true",
    "endpoints": [
        "http:\/\/rgw1:80"
    ],
    "hostnames": [],
    "hostnames_s3website": [],
    "master_zone": "9248cab2-afe7-43d8-a661-a40bf316665e",
    "zones": [
        {
            "id": "9248cab2-afe7-43d8-a661-a40bf316665e",
            "name": "us-east",
            "endpoints": [
                "http:\/\/rgw1"
            ],
            "log_meta": "true",
            "log_data": "true",
            "bucket_index_max_shards": 11,
```

```
            "read_only": "false"
        },
        {
            "id": "d1024e59-7d28-49d1-8222-af101965a939",
            "name": "us-west",
            "endpoints": [
                "http:\/\/rgw2:80"
            ],
            "log_meta": "false",
            "log_data": "true",
            "bucket_index_max_shards": 11,
            "read_only": "false"
        }
    ],
    "placement_targets": [
        {
            "name": "default-placement",
            "tags": []
        }
    ],
    "default_placement": "default-placement",
    "realm_id": "ae031368-8715-4e27-9a99-0c9468852cfe"
}
```

5.11.2.9. ゾーングループの設定

ゾーングループの定義は、少なくとも必要な設定を指定して JSON オブジェクトの作成で設定されます。

1. **name**: ゾーングループの名前。必須です。
2. **api_name**: ゾーングループの API 名。任意です。
3. **is_master**: ゾーングループがマスターゾーングループであるかどうかを指定します。必須です。
   注記: マスターゾーングループを 1 つだけ指定できます。
4. **endpoints**: ゾーングループ内のエンドポイントのリスト。たとえば、複数のドメイン名を使用して、同じゾーングループを参照できます。忘れずに前方スラッシュ (\/) エスケープしてください。各エンドポイントにポート (**fqdn:port**) を指定することもできます。任意です。
5. **hostnames**: ゾーングループのホスト名のリスト。たとえば、複数のドメイン名を使用して、同じゾーングループを参照できます。任意です。**rgw dns name** 設定は、このリストに自動的に含まれます。この設定を変更したら、ゲートウェイデーモンを再起動する必要があります。
6. **master_zone**: ゾーングループのマスターゾーン。任意です。指定がない場合は、デフォルトゾーンを使用します。

   注記

   ゾーングループごとにマスターゾーンを 1 つだけ指定できます。

7. **zones**: ゾーングループ内のゾーンのリスト。各ゾーンには、名前 (必須)、エンドポイントのリスト (任意)、およびゲートウェイがメタデータおよびデータ操作をログに記録するかどうか (デフォルトでは false) があります。

8. **placement_targets**: 配置ターゲットのリスト (任意)。各配置ターゲットには、配置ターゲットの名前 (必須) とタグのリスト (任意) が含まれているため、タグを持つユーザーのみが配置ターゲットを使用できます (つまり、ユーザー情報のユーザーの **placement_tags** フィールド)。

9. **default_placement**: オブジェクトインデックスおよびオブジェクトデータのデフォルトの配置ターゲット。デフォルトでは **default-placement** に設定されます。また、ユーザーごとのデフォルトの配置を、ユーザー情報で設定することもできます。

ゾーングループを設定するには、必須フィールドで設定される JSON オブジェクトを作成し、オブジェクトをファイル (たとえば、**zonegroup.json**) に保存します。次に、次のコマンドを実行します。

例

```
[ceph: root@host01 /]# radosgw-admin zonegroup set --infile zonegroup.json
```

ここで、**zonegroup.json** は作成した JSON ファイルです。

重要

default ゾーングループの **is_master** 設定は **true** です。新しいゾーングループを作成してそれをマスターゾーングループにしたい場合は、**default** ゾーングループ **is_master** 設定を **false** に設定するか、**default** ゾーングループを削除する必要があります。

最後に、期間を更新します。

例

```
[ceph: root@host01 /]# radosgw-admin period update --commit
```

5.11.2.10. ゾーングループマップの設定

ゾーングループマップの設定は、1 つ以上のゾーングループで設定される JSON オブジェクトの作成と、クラスターの **master_zonegroup の** 設定で設定されます。ゾーングループマップの各ゾーングループは、キーと値のペアで設定されます。**key** 設定は、個々のゾーングループ設定の **名前** 設定と同等であり、**val** は、個々のゾーングループ設定で設定される JSON オブジェクトです。

is_master が **true** と同等のゾーングループを 1 つだけ持つ可能性があり、ゾーングループマップの最後に **master_zonegroup** として指定する必要があります。以下の JSON オブジェクトは、デフォルトゾーングループマップの例です。

```
{
    "zonegroups": [
        {
            "key": "90b28698-e7c3-462c-a42d-4aa780d24eda",
            "val": {
                "id": "90b28698-e7c3-462c-a42d-4aa780d24eda",
                "name": "us",
                "api_name": "us",
                "is_master": "true",
                "endpoints": [
                    "http:\/\/rgw1:80"
                ],
                "hostnames": [],
                "hostnames_s3website": [],
```

```
            "master_zone": "9248cab2-afe7-43d8-a661-a40bf316665e",
            "zones": [
                {
                    "id": "9248cab2-afe7-43d8-a661-a40bf316665e",
                    "name": "us-east",
                    "endpoints": [
                        "http:\/\/rgw1"
                    ],
                    "log_meta": "true",
                    "log_data": "true",
                    "bucket_index_max_shards": 11,
                    "read_only": "false"
                },
                {
                    "id": "d1024e59-7d28-49d1-8222-af101965a939",
                    "name": "us-west",
                    "endpoints": [
                        "http:\/\/rgw2:80"
                    ],
                    "log_meta": "false",
                    "log_data": "true",
                    "bucket_index_max_shards": 11,
                    "read_only": "false"
                }
            ],
            "placement_targets": [
                {
                    "name": "default-placement",
                    "tags": []
                }
            ],
            "default_placement": "default-placement",
            "realm_id": "ae031368-8715-4e27-9a99-0c9468852cfe"
        }
    ],
    "master_zonegroup": "90b28698-e7c3-462c-a42d-4aa780d24eda",
    "bucket_quota": {
        "enabled": false,
        "max_size_kb": -1,
        "max_objects": -1
    },
    "user_quota": {
        "enabled": false,
        "max_size_kb": -1,
        "max_objects": -1
    }
}
```

ゾーングループマップを設定するには、次のコマンドを実行します。

例

```
[ceph: root@host01 /]# radosgw-admin zonegroup-map set --infile zonegroupmap.json
```

ここで、**zonegroupmap.json** は作成した JSON ファイルです。ゾーングループマップで指定したゾーンが作成されていることを確認します。最後に、期間を更新します。

例

```
[ceph: root@host01 /]#  radosgw-admin period update --commit
```

5.11.3. ゾーン

Ceph Object Gateway はゾーンの概念をサポートします。ゾーンは、1 つ以上の Ceph Object Gateway インスタンスで設定される論理グループを定義します。

ゾーンの設定は、Ceph 設定ファイル内で終了するすべての設定ではないため、一般的な設定手順とは異なります。ゾーンをリスト表示して、ゾーン設定を取得し、ゾーン設定を設定できます。

重要

radosgw-admin zone 操作はすべて、ゾーン内で動作するまたはこれから動作するホストで発行する **必要があります**。

5.11.3.1. ゾーンの作成

ゾーンを作成するには、ゾーン名を指定します。マスターゾーンの場合は、**--master** オプションを指定します。ゾーングループ内の 1 つのゾーンのみがマスターゾーンになることができます。ゾーングループにゾーンを追加するには、**--rgw-zonegroup** オプションをゾーングループ名で指定します。

重要

ゾーン内の Ceph Object Gateway ノードでゾーンを作成する必要があります。

構文

```
radosgw-admin zone create --rgw-zone=ZONE_NAME \
             [--zonegroup=ZONE_GROUP_NAME]\
             [--endpoints=ENDPOINT_PORT [,<endpoint:port>] \
             [--master] [--default] \
             --access-key ACCESS_KEY --secret SECRET_KEY
```

次に、期間を更新します。

例

```
[ceph: root@host01 /]# radosgw-admin period update --commit
```

5.11.3.2. ゾーンの削除

ゾーンを削除するには、最初にゾーングループからこれを削除します。

手順

1. ゾーングループからゾーンを削除します。

構文

```
radosgw-admin zonegroup remove --rgw-zonegroup=ZONE_GROUP_NAME\
                               --rgw-zone=ZONE_NAME
```

2. 期間を更新します。

例

```
[ceph: root@host01 /]# radosgw-admin period update --commit
```

3. ゾーンを削除します。

重要

この手順は、ゾーン内のホストで**必ず**使用する必要があります。

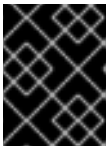
構文

```
radosgw-admin zone delete --rgw-zone=ZONE_NAME
```

4. 期間を更新します。

例

```
[ceph: root@host01 /]# radosgw-admin period update --commit
```



重要

ゾーングループから先にゾーンを削除せずに、ゾーンを削除しないでください。それ以外の場合には、期間の更新に失敗します。

削除したゾーンのプールが他に使用されていない場合は、プールを削除することを検討してください。以下の例の **DELETED_ZONE_NAME** を、削除したゾーン名に置き換えます。

重要

Ceph がゾーンプールを削除すると、それによってリカバリー不可能な方法でその中のデータが削除されます。Ceph クライアントにプールの内容が必要なくなった場合にのみ、ゾーンプールを削除します。

重要

マルチレルムクラスターでは、**.rgw.root** プールをゾーンプールと共に削除すると、クラスターのレルム情報のすべてが削除されます。**.rgw.root** プールを削除する前に、**.rgw.root** に他のアクティブなレルムが含まれていないことを確認します。

構文

```
ceph osd pool delete DELETED_ZONE_NAME.rgw.control DELETED_ZONE_NAME.rgw.control --
yes-i-really-really-mean-it
```

```
ceph osd pool delete DELETED_ZONE_NAME.rgw.data.root
DELETED_ZONE_NAME.rgw.data.root --yes-i-really-really-mean-it
ceph osd pool delete DELETED_ZONE_NAME.rgw.log DELETED_ZONE_NAME.rgw.log --yes-i-
really-really-mean-it
ceph osd pool delete DELETED_ZONE_NAME.rgw.users.uid
DELETED_ZONE_NAME.rgw.users.uid --yes-i-really-really-mean-it
```

重要

プールの削除後に、RGW プロセスを再起動します。

5.11.3.3. ゾーンの変更

ゾーンを変更するには、ゾーン名と、変更するパラメーターを指定します。

重要

ゾーンは、ゾーン内にある Ceph Object Gateway ノードで変更する必要があります。

構文

```
radosgw-admin zone modify [options]

--access-key=<key>
--secret/--secret-key=<key>
--master
--default
--endpoints=<list>
```

次に、期間を更新します。

例

```
[ceph: root@host01 /]# radosgw-admin period update --commit
```

5.11.3.4. ゾーンのリスト

root でクラスター内のゾーンをリスト表示するには、以下のコマンドを実行します。

例

```
[ceph: root@host01 /]# radosgw-admin zone list
```

5.11.3.5. ゾーンの取得

root でゾーンの設定を取得するには、次のコマンドを実行します。

構文

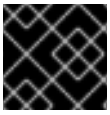
```
radosgw-admin zone get [--rgw-zone=ZONE_NAME]
```

default ゾーンは以下のようになります。

```
{ "domain_root": ".rgw",
  "control_pool": ".rgw.control",
  "gc_pool": ".rgw.gc",
  "log_pool": ".log",
  "intent_log_pool": ".intent-log",
  "usage_log_pool": ".usage",
  "user_keys_pool": ".users",
  "user_email_pool": ".users.email",
  "user_swift_pool": ".users.swift",
  "user_uid_pool": ".users.uid",
  "system_key": { "access_key": "", "secret_key": ""},
  "placement_pools": [
     {  "key": "default-placement",
        "val": { "index_pool": ".rgw.buckets.index",
                 "data_pool": ".rgw.buckets"}
     }
   ]
  }
```

5.11.3.6. ゾーンの設定

ゾーンの設定には、一連の Ceph Object Gateway プールを指定する必要があります。一貫性を保つために、ゾーン名と同じプールの接頭辞を使用することが推奨されます。プールの設定に関する詳細は、**Red Hat Ceph Storage Storage ストラテジーガイド**の プール の章を参照してください。



重要

ゾーン内の Ceph Object Gateway ノードでゾーンを設定する必要があります。

ゾーンを設定するには、プールで設定される JSON オブジェクトを作成し、オブジェクトをファイル (例: **zone.json**) に保存します。続いて以下のコマンドを実行して、**ZONE_NAME** をゾーンの名前に置き換えます。

例

```
[ceph: root@host01 /]# radosgw-admin zone set --rgw-zone=test-zone --infile zone.json
```

ここで、**zone.json** は作成した JSON ファイルです。

次に、**root** でピリオドを更新します。

例

```
[ceph: root@host01 /]# radosgw-admin period update --commit
```

5.11.3.7. ゾーンの名前変更

ゾーンの名前を変更するには、ゾーン名および新しいゾーン名を指定します。ゾーン内のホストで以下のコマンドを発行します。

構文

```
radosgw-admin zone rename --rgw-zone=ZONE_NAME --zone-new-name=NEW_ZONE_NAME
```

次に、期間を更新します。

例

```
[ceph: root@host01 /]# radosgw-admin period update --commit
```

第6章 詳細設定

ストレージ管理者は、Ceph Object Gateway の高度な機能の一部を設定できます。マルチサイト Ceph Object Gateway を設定し、Microsoft Active Directory サービスや OpenStack Keystone サービスなどのディレクトリーサービスと統合することができます。

前提条件

- 正常かつ実行中の Red Hat Ceph Storage クラスター

6.1. LDAP および CEPH OBJECT GATEWAY の設定

Ceph Object Gateway ユーザーを認証するように Red Hat Directory Server を設定するには、以下の手順を実施します。

6.1.1. Red Hat Directory Server のインストール

Java Swing GUI Directory および管理コンソールを使用するには、グラフィカルユーザーインターフェイス (GUI) を使用する Red Hat Enterprise Linux 9 に Red Hat Directory Server をインストールする必要があります。ただし、Red Hat Directory Server にはコマンドラインインターフェイス (CLI) から排他的にサービスを提供できます。

前提条件

- Red Hat Enterprise Linux (RHEL) がサーバーにインストールされている。
- Directory Server ノードの FQDN は、DNS または **/etc/hosts** ファイルを使用して解決可能です。
- Directory Server ノードを Red Hat サブスクリプション管理サービスに登録します。
- お使いの Red Hat アカウントに有効な Red Hat Directory Server サブスクリプションが利用できます。

手順

- **Red Hat Directory Server インストールガイド**の 第 1 章 および 第 2 章 の指示に従ってください。

関連情報

- 詳細は、Red Hat Director Server インストールガイド を参照してください。

6.1.2. Directory Server ファイアウォールの設定

LDAP ホストで、LDAP クライアントが Directory Server にアクセスできるように、ファイアウォールが Directory Server のセキュアな (**636**) ポートにアクセスできることを確認します。デフォルトのセキュアでないポート (**389**) を閉じたままにしておきます。

```
# firewall-cmd --zone=public --add-port=636/tcp
# firewall-cmd --zone=public --add-port=636/tcp --permanent
```

6.1.3. SELinux のラベルポート

SELinux が要求をブロックしないようにするには、SELinux のポートにラベルを付けます。詳細は、Red Hat Directory Server 10 の**Administration Guide**の Changing Directory Server Port Numbers を参照してください。

6.1.4. LDAPS の設定

Ceph Object Gateway は単純な ID およびパスワードを使用して LDAP サーバーとの認証を行うため、接続には LDAP の SSL 証明書が必要です。LDAP 用 Directory Server を設定するには、Red Hat Directory Server 11 の **Administration Guide** で Configuring Secure Connections の章を参照してください。

LDAP が動作したら、Ceph Object Gateway サーバーが Directory Server の証明書を信頼するように設定します。

1. LDAP サーバーの SSL 証明書に署名した認証局 (CA) の PEM 形式の証明書を抽出/ダウンロードします。
2. **/etc/openldap/ldap.conf** に **TLS_REQCERT** が設定されていないことを確認します。
3. **/etc/openldap/ldap.conf** に **TLS_CACERTDIR /etc/openldap/certs** 設定が含まれていることを確認します。
4. **certutil** コマンドを使用して、AD CA を **/etc/openldap/certs** のストアに追加します。たとえば、CA が msad-frog-MSAD-FROG-CA で、PEM 形式の CA ファイルが **ldap.pem** の場合は、以下のコマンドを使用します。

   例

   ```
   # certutil -d /etc/openldap/certs -A -t "TC,," -n "msad-frog-MSAD-FROG-CA" -i
   /path/to/ldap.pem
   ```

5. すべてのリモート LDAP サイトで SELinux を更新します。

   例

   ```
   # setsebool -P httpd_can_network_connect on
   ```

   注記

   これは、SELinux が Permissive モードであっても、引き続き設定する必要があります。

6. **certs** データベースを誰でも読めるようにします。

   例

   ```
   # chmod 644 /etc/openldap/certs/*
   ```

7. root 以外のユーザーとして ldapwhoami コマンドを使用してサーバーに接続します。

   例

   ```
   $ ldapwhoami -H ldaps://rh-directory-server.example.com -d 9
   ```

-d 9 オプションは、SSL ネゴシエーションで何らかの問題が発生した場合にデバッグ情報を提供します。

6.1.5. ゲートウェイユーザーの有無の確認

ゲートウェイユーザーを作成する前に、Ceph Object Gateway にユーザーがまだ存在していないことを確認してください。

例

```
[ceph: root@host01 /]# radosgw-admin metadata list user
```

このユーザー名は、このユーザーリストには記載しないでください。

6.1.6. ゲートウェイユーザーの追加

LDAP ユーザーへの Ceph Object Gateway ユーザーを作成します。

手順

1. Ceph Object Gateway の LDAP ユーザーを作成し、**binddn** を書き留めます。Ceph オブジェクトゲートウェイは **ceph** ユーザーを使用するため、**ceph** をユーザー名として使用することを検討してください。ユーザーに、ディレクトリーを検索するパーミッションが必要です。Ceph Object Gateway は、**rgw_ldap_binddn** で指定されたとおりにこのユーザーにバインドします。

2. ユーザーの作成が正常に機能することをテストします。**ceph** が **People** の下のユーザー ID で、**example.com** がドメインの場合は、ユーザーの検索を行うことができます。

   ```
   # ldapsearch -x -D "uid=ceph,ou=People,dc=example,dc=com" -W -H ldaps://example.com -b "ou=People,dc=example,dc=com" -s sub 'uid=ceph'
   ```

3. 各ゲートウェイノードで、ユーザーのシークレットのファイルを作成します。たとえば、シークレットは **/etc/bindpass** という名前のファイルに保存されます。セキュリティー上の理由から、このファイルの所有者を **ceph** ユーザーおよびグループに変更し、グローバルに読み取りができないようにします。

4. **rgw_ldap_secret** オプションを追加します。

   構文

   ```
   ceph config set client.rgw OPTION VALUE
   ```

   例

   ```
   [ceph: root@host01 /]# ceph config set client.rgw rgw_ldap_secret /etc/bindpass
   ```

5. バインドパスワードファイルを Ceph Object Gateway コンテナーにパッチし、Ceph Object Gateway 仕様を再適用します。

   例

   ```
   service_type: rgw
   ```

```
service_id: rgw.1
service_name: rgw.rgw.1
placement:
  label: rgw
  extra_container_args:
  - -v
  - /etc/bindpass:/etc/bindpass
```

注記

/etc/bindpass は Red Hat Ceph Storage に自動的に同梱されないため、すべての Ceph Object Gateway インスタンスノードでコンテンツが利用可能であることを確認する必要があります。

6.1.7. LDAP を使用するようにゲートウェイを設定

1. すべての Ceph ノードで次のコマンドを使用して Ceph 設定を変更します。

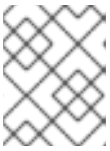
   構文

   ```
   ceph config set client.rgw OPTION VALUE
   ```

   例

   ```
   [ceph: root@host01 /]# ceph config set client.rgw rgw_ldap_uri ldaps://:636
   [ceph: root@host01 /]# ceph config set client.rgw rgw_ldap_binddn
   "ou=poc,dc=example,dc=local"
   [ceph: root@host01 /]# ceph config set client.rgw rgw_ldap_searchdn
   "ou=poc,dc=example,dc=local"
   [ceph: root@host01 /]# ceph config set client.rgw rgw_ldap_dnattr "uid"
   [ceph: root@host01 /]# ceph config set client.rgw rgw_s3_auth_use_ldap true
   ```

2. Ceph Object Gateway を再起動します。

   注記

   NAME 列の **ceph orch ps** コマンドの出力を使用して、**SERVICE_TYPE.ID** 情報を取得します。

   a. ストレージクラスター内の個別のノードで Ceph Object Gateway を再起動するには、以下を実行します。

      構文

      ```
      systemctl restart ceph-CLUSTER_ID@SERVICE_TYPE.ID.service
      ```

      例

      ```
      [root@host01 ~]# systemctl restart ceph-c4b34c6f-8365-11ba-dc31-
      529020a7702d@rgw.realm.zone.host01.gwasto.service
      ```

b. ストレージクラスター内のすべてのノードで Ceph Object Gateway を再起動するには、以下を実行します。

構文

```
ceph orch restart SERVICE_TYPE
```

例

```
[ceph: root@host01 /]# ceph orch restart rgw
```

6.1.8. カスタム検索フィルターの使用

rgw_ldap_searchfilter 設定を使用すると、ユーザーアクセスを制限するカスタム検索フィルターを作成できます。**rgw_ldap_searchfilter** 設定には、2 つの方法があります。

1. 部分フィルターの指定

   例

   ```
   "objectclass=inetorgperson"
   ```

   Ceph Object Gateway は、トークンのユーザー名および **rgw_ldap_dnattr** の値を使用して検索フィルターを生成します。構築されたフィルターは、**rgw_ldap_searchfilter** の値の一部フィルターと組み合わされます。たとえば、ユーザー名と設定により、最終的な検索フィルターが生成されます。

   例

   ```
   "(&(uid=joe)(objectclass=inetorgperson))"
   ```

   ユーザー **joe** は、LDAP ディレクトリーで見つかった場合にのみアクセスが許可され、**inetorgperson** のオブジェクトクラスがあり、有効なパスワードを指定します。

2. Complete フィルターの指定
   完全なフィルターには、認証の試行中にユーザー名に置き換えられる **USERNAME** トークンが含まれている必要があります。この場合、**rgw_ldap_dnattr** 設定は使用されません。たとえば、有効なユーザーを特定のグループに制限するには、以下のフィルターを使用します。

   例

   ```
   "(&(uid=@USERNAME@)(memberOf=cn=ceph-users,ou=groups,dc=mycompany,dc=com))"
   ```

6.1.9. S3 ユーザーの LDAP サーバーへの追加

LDAP サーバーの管理コンソールで S3 ユーザーを少なくとも 1 つ作成し、S3 クライアントが LDAP ユーザーの認証情報を使用できるようにします。認証情報を S3 クライアントに渡すときに使用するユーザー名およびシークレットを書き留めておきます。

6.1.10. LDAP トークンのエクスポート

LDAP で Ceph Object Gateway を実行する場合は、アクセストークンのみが必要です。ただし、アクセストークンは、アクセスキーとシークレットキーから作成されます。アクセスキーとシークレットキーを LDAP トークンとしてエクスポートします。

1. アクセスキーをエクスポートします。

   構文

   ```
   export RGW_ACCESS_KEY_ID="USERNAME"
   ```

2. シークレットキーをエクスポートします。

   構文

   ```
   export RGW_SECRET_ACCESS_KEY="PASSWORD"
   ```

3. トークンをエクスポートします。LDAP の場合は、トークンタイプ (**ttype**) に **ldap** を使用します。

   例

   ```
   radosgw-token --encode --ttype=ldap
   ```

   Active Directory の場合は、トークンタイプとして **ad** を使用します。

   例

   ```
   radosgw-token --encode --ttype=ad
   ```

   結果として、アクセストークンである base-64 でエンコードされた文字列になります。このアクセストークンを、アクセスキーの代わりに S3 クライアントに提供します。シークレットキーは不要になりました。

4. オプション:S3 クライアントが環境変数を使用している場合は、利便性を高めるために base-64 でエンコードされた文字列を環境変数 **RGW_ACCESS_KEY_ID** にエクスポートします。

   例

   ```
   export
   RGW_ACCESS_KEY_ID="ewogICAgIlJHV19UT0tFTiI6IHsKICAgICAgICAidmVyc2lvbiI6IDEsCiAgICAgICAgInR5cGUiOiAibGRhcCIsCiAgICAgICAgImlkIjogImNlcGgiLAogICAgICAgICJrZXkiOiAiODAwI0dvcmlsbGEiCiAgICB9Cn0K"
   ```

6.1.11. S3 クライアントを使用した設定のテスト

Python Boto などのスクリプトを使用して、Ceph Object Gateway クライアントで設定をテストします。

手順

1. **RGW_ACCESS_KEY_ID** 環境変数を使用して、Ceph Object Gateway クライアントを設定します。または、base-64 でエンコードされた文字列をコピーし、アクセスキーとして指定することもできます。以下は、設定された S3 クライアントの例です。

例

```
cat .aws/credentials

[default]
aws_access_key_id =
ewogICaGbnjlwe9UT0tFTiI6IHsKICAgICAgICAidmVyc2lvbiI6IDEsCiAgICAgICAgInR5cGUiOiAi
YWQiLAogICAgICAgICJpZCI6ICJjZXBoIiwKICAgICAgICAia2V5IjogInBhc3M0Q2VwaCIKICAgI
H0KfQo=
aws_secret_access_key =
```

注記

シークレットキーは不要になりました。

2. **aws s3 ls** コマンドを実行してユーザーを確認します。

例

```
[root@host01 ~]# aws s3 ls --endpoint http://host03

2023-12-11 17:08:50 mybucket
2023-12-24 14:55:44 mybucket2
```

3. オプション: **radosgw-admin user** コマンドを実行して、ディレクトリー内のユーザーを確認することもできます。

例

```
[root@host01 ~]# radosgw-admin user info --uid dir1
{
    "user_id": "dir1",
    "display_name": "dir1",
    "email": "",
    "suspended": 0,
    "max_buckets": 1000,
    "subusers": [],
    "keys": [],
    "swift_keys": [],
    "caps": [],
    "op_mask": "read, write, delete",
    "default_placement": "",
    "default_storage_class": "",
    "placement_tags": [],
    "bucket_quota": {
        "enabled": false,
        "check_on_raw": false,
        "max_size": -1,
        "max_size_kb": 0,
        "max_objects": -1
    },
    "user_quota": {
        "enabled": false,
        "check_on_raw": false,
```

```
        "max_size": -1,
        "max_size_kb": 0,
        "max_objects": -1
    },
    "temp_url_keys": [],
    "type": "ldap",
    "mfa_ids": []
}
```

6.2. ACTIVE DIRECTORY および CEPH OBJECT GATEWAY の設定

Ceph Object Gateway ユーザーを認証するように Active Directory サーバーを設定するには、以下の手順を実施します。

6.2.1. Microsoft Active Directory の使用

Ceph Object Gateway の LDAP 認証は、Microsoft Active Directory を含む単純なバインド用に設定できる LDAP 準拠のディレクトリーサービスと互換性があります。Active Directory の使用は、Ceph Object Gateway が **rgw_ldap_binddn** 設定に設定されたユーザーとしてバインドし、LDAP を使用してセキュリティーを確保する RH Directory サーバーの使用と似ています。

Active Directory を設定するプロセスは、基本的に LDAP および Ceph Object Gateway を設定するプロセスと同じですが、Windows 固有の使用法がいくつかある場合があります。

6.2.2. LDAPS の Active Directory の設定

Active Directory LDAP サーバーは、デフォルトで LDAP を使用するように設定されています。Windows Server 2012 以降では、Active Directory 証明書サービスを使用できます。Active Directory LDAP で使用する SSL 証明書を生成してインストールする手順は、MS TechNet の記事 LDAP over SSL (LDAPS) Certificate を参照してください。

注記

ポート **636** が Active Directory ホストで開いていることを確認します。

6.2.3. ゲートウェイユーザーの有無の確認

ゲートウェイユーザーを作成する前に、Ceph Object Gateway にユーザーがまだ存在していないことを確認してください。

例

```
[ceph: root@host01 /]# radosgw-admin metadata list user
```

このユーザー名は、このユーザーリストには記載しないでください。

6.2.4. ゲートウェイユーザーの追加

LDAP ユーザーへの Ceph Object Gateway ユーザーを作成します。

手順

1. Ceph Object Gateway の LDAP ユーザーを作成し、**binddn** を書き留めます。Ceph オブジェ

クトゲートウェイは **ceph** ユーザーを使用するため、**ceph** をユーザー名として使用することを検討してください。ユーザーに、ディレクトリーを検索するパーミッションが必要です。Ceph Object Gateway は、**rgw_ldap_binddn** で指定されたとおりにこのユーザーにバインドします。

2. ユーザーの作成が正常に機能することをテストします。**ceph** が **People** の下のユーザー ID で、**example.com** がドメインの場合は、ユーザーの検索を行うことができます。

```
# ldapsearch -x -D "uid=ceph,ou=People,dc=example,dc=com" -W -H ldaps://example.com -b "ou=People,dc=example,dc=com" -s sub 'uid=ceph'
```

3. 各ゲートウェイノードで、ユーザーのシークレットのファイルを作成します。たとえば、シークレットは **/etc/bindpass** という名前のファイルに保存されます。セキュリティー上の理由から、このファイルの所有者を **ceph** ユーザーおよびグループに変更し、グローバルに読み取りができないようにします。

4. **rgw_ldap_secret** オプションを追加します。

構文

```
ceph config set client.rgw OPTION VALUE
```

例

```
[ceph: root@host01 /]# ceph config set client.rgw rgw_ldap_secret /etc/bindpass
```

5. バインドパスワードファイルを Ceph Object Gateway コンテナーにパッチし、Ceph Object Gateway 仕様を再適用します。

例

```
service_type: rgw
service_id: rgw.1
service_name: rgw.rgw.1
placement:
  label: rgw
  extra_container_args:
  - -v
  - /etc/bindpass:/etc/bindpass
```

注記

/etc/bindpass は Red Hat Ceph Storage に自動的に同梱されないため、すべての Ceph Object Gateway インスタンスノードでコンテンツが利用可能であることを確認する必要があります。

6.2.5. Active Directory を使用するようにゲートウェイを設定

1. **rgw_ldap_secret** の設定後に、以下のオプションを追加します。

構文

```
ceph config set client.rgw OPTION VALUE
```

例

```
[ceph: root@host01 /]# ceph config set client.rgw rgw_ldap_uri ldaps://_FQDN_:636
[ceph: root@host01 /]# ceph config set client.rgw rgw_ldap_binddn "_BINDDN_"
[ceph: root@host01 /]# ceph config set client.rgw rgw_ldap_searchdn "_SEARCHDN_"
[ceph: root@host01 /]# ceph config set client.rgw rgw_ldap_dnattr "cn"
[ceph: root@host01 /]# ceph config set client.rgw rgw_s3_auth_use_ldap true
```

rgw_ldap_uri 設定の場合は、**FQDN** を LDAP サーバーの完全修飾ドメイン名に置き換えます。複数の LDAP サーバーがある場合には、各ドメインを指定します。

rgw_ldap_binddn 設定の場合は、**BINDDN** をバインドドメインに置き換えます。**users** および **accounts** の下に **example.com** のドメインおよび **ceph** ユーザーが使用されている場合には、以下のようになります。

例

```
rgw_ldap_binddn "uid=ceph,cn=users,cn=accounts,dc=example,dc=com"
```

rgw_ldap_searchdn 設定の場合は、**SEARCHDN** を検索ドメインに置き換えます。**users** および **accounts** の下に **example.com** のドメインおよびユーザーがある場合は、以下のようになります。

```
rgw_ldap_searchdn "cn=users,cn=accounts,dc=example,dc=com"
```

2. Ceph Object Gateway を再起動します。

注記

NAME 列の **ceph orch ps** コマンドの出力を使用して、**SERVICE_TYPE.ID** 情報を取得します。

a. ストレージクラスター内の個別のノードで Ceph Object Gateway を再起動するには、以下を実行します。

構文

```
systemctl restart ceph-CLUSTER_ID@SERVICE_TYPE.ID.service
```

例

```
[root@host01 ~]# systemctl restart ceph-c4b34c6f-8365-11ba-dc31-
529020a7702d@rgw.realm.zone.host01.gwasto.service
```

b. ストレージクラスター内のすべてのノードで Ceph Object Gateway を再起動するには、以下を実行します。

構文

```
ceph orch restart SERVICE_TYPE
```

例

```
[ceph: root@host01 /]# ceph orch restart rgw
```

6.2.6. S3 ユーザーの LDAP サーバーへの追加

LDAP サーバーの管理コンソールで S3 ユーザーを少なくとも 1 つ作成し、S3 クライアントが LDAP ユーザーの認証情報を使用できるようにします。認証情報を S3 クライアントに渡すときに使用するユーザー名およびシークレットを書き留めておきます。

6.2.7. LDAP トークンのエクスポート

LDAP で Ceph Object Gateway を実行する場合は、アクセストークンのみが必要です。ただし、アクセストークンは、アクセスキーとシークレットキーから作成されます。アクセスキーとシークレットキーを LDAP トークンとしてエクスポートします。

1. アクセスキーをエクスポートします。

   構文

   ```
   export RGW_ACCESS_KEY_ID="USERNAME"
   ```

2. シークレットキーをエクスポートします。

   構文

   ```
   export RGW_SECRET_ACCESS_KEY="PASSWORD"
   ```

3. トークンをエクスポートします。LDAP の場合は、トークンタイプ (**ttype**) に **ldap** を使用します。

   例

   ```
   radosgw-token --encode --ttype=ldap
   ```

   Active Directory の場合は、トークンタイプとして **ad** を使用します。

   例

   ```
   radosgw-token --encode --ttype=ad
   ```

   結果として、アクセストークンである base-64 でエンコードされた文字列になります。このアクセストークンを、アクセスキーの代わりに S3 クライアントに提供します。シークレットキーは不要になりました。

4. オプション:S3 クライアントが環境変数を使用している場合は、利便性を高めるために base-64 でエンコードされた文字列を環境変数 **RGW_ACCESS_KEY_ID** にエクスポートします。

   例

   ```
   export
   RGW_ACCESS_KEY_ID="ewogICAgIlJHV19UT0tFTiI6IHsKICAgICAgICAidmVyc2lvbiI6IDEsCiAgICAgICAgInR5cGUiOiAibGRhcCIsCiAgICAgICAgImlkIjogImNlcGgiLAogICAgICAgICJrZXkiOiAiODAwI0dvcmlsbGEiCiAgICB9Cn0K"
   ```

6.2.8. S3 クライアントを使用した設定のテスト

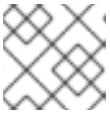
Python Boto などのスクリプトを使用して、Ceph Object Gateway クライアントで設定をテストします。

手順

1. **RGW_ACCESS_KEY_ID** 環境変数を使用して、Ceph Object Gateway クライアントを設定します。または、base-64 でエンコードされた文字列をコピーし、アクセスキーとして指定することもできます。以下は、設定された S3 クライアントの例です。

   例

   ```
   cat .aws/credentials

   [default]
   aws_access_key_id =
   ewogICaGbnjlwe9UT0tFTiI6IHsKICAgICAgICAidmVyc2lvbiI6IDEsCiAgICAgICAgInR5cGUiOiAi
   YWQiLAogICAgICAgICJpZCI6ICJjZXBoIiwKICAgICAgICAia2V5IjogInBhc3M0Q2VwaCIKICAgI
   H0KfQo=
   aws_secret_access_key =
   ```

   注記

   シークレットキーは不要になりました。

2. **aws s3 ls** コマンドを実行してユーザーを確認します。

   例

   ```
   [root@host01 ~]# aws s3 ls --endpoint http://host03

   2023-12-11 17:08:50 mybucket
   2023-12-24 14:55:44 mybucket2
   ```

3. オプション: **radosgw-admin user** コマンドを実行して、ディレクトリー内のユーザーを確認することもできます。

   例

   ```
   [root@host01 ~]# radosgw-admin user info --uid dir1
   {
       "user_id": "dir1",
       "display_name": "dir1",
       "email": "",
       "suspended": 0,
       "max_buckets": 1000,
       "subusers": [],
       "keys": [],
       "swift_keys": [],
       "caps": [],
       "op_mask": "read, write, delete",
       "default_placement": "",
       "default_storage_class": "",
   ```

```
    "placement_tags": [],
    "bucket_quota": {
        "enabled": false,
        "check_on_raw": false,
        "max_size": -1,
        "max_size_kb": 0,
        "max_objects": -1
    },
    "user_quota": {
        "enabled": false,
        "check_on_raw": false,
        "max_size": -1,
        "max_size_kb": 0,
        "max_objects": -1
    },
    "temp_url_keys": [],
    "type": "ldap",
    "mfa_ids": []
}
```


6.3. CEPH OBJECT GATEWAY および OPENSTACK KEYSTONE

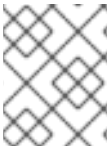
ストレージ管理者は、OpenStack の Keystone 認証サービスを使用して、Ceph Object Gateway 経由でユーザーを認証することができます。Ceph Object Gateway を設定する前に、最初に Keystone を設定する必要があります。これにより、Swift サービスが有効になり、Keystone サービスが Ceph Object Gateway を指すようになります。次に、Ceph Object Gateway が Keystone サービスからの認証要求を受け入れるように設定する必要があります。

前提条件

- 実行中の Red Hat OpenStack Platform 環境
- 稼働中の Red Hat Ceph Storage 環境
- 実行中の Ceph Object Gateway 環境。

6.3.1. Keystone 認証用のロール

OpenStack Keystone サービスは、**admin**、**member**、および **reader** の 3 つのロールを提供します。これらのロールは階層的です。**admin** ロールを持つユーザーは、**member** ロールの機能を、**member** ロールを持つユーザーは **reader** ロールの機能を継承します。



注記

member ロールの読み取りパーミッションは、所属するプロジェクトのオブジェクトにのみ適用されます。

admin

admin ロールは、特定のスコープ内で最も高い承認レベルとして予約されます。これには通常、リソースまたは API のすべての作成、読み取り、更新、または削除操作が含まれます。

member

member ロールは、デフォルトでは直接使用されません。柔軟にデプロイメントでき、管理者の責務を軽減するのに役立ちます。

たとえば、デフォルトの **member** ロールと単純なポリシーオーバーライドを使用してデプロイメントのポリシーを上書きし、system member がサービスとエンドポイントを更新できるようにすることができます。これにより、**admin** と **reader** ロール間の承認の階層ができます。

reader

reader ロールは、スコープに関係なく読み取り専用の操作向けに予約されます。

警告

reader を使用してイメージライセンスキー、管理イメージデータ、管理ボリュームメタデータ、アプリケーション認証情報、シークレットなどの機密情報にアクセスする場合は、意図せずに機密情報を公開する可能性があります。そのため、これらのリソースを公開する API は、**reader** ロールの影響を慎重に考慮し、**member** および **admin** ロールへの適切なアクセスを継承する必要があります。

6.3.2. Keystone 認証および Ceph Object Gateway

OpenStack Keystone を使用してユーザーを認証する組織では、Keystone と Ceph Object Gateway を統合することができます。Ceph Object Gateway は、ゲートウェイが Keystone トークンを受け入れ、ユーザーを認証して対応する Ceph Object Gateway ユーザーを作成できるようにします。Keystone がトークンを検証すると、ゲートウェイはユーザーが認証されたを見なします。

利点

- Keystone を持つユーザーに **admin**、**member**、および **reader** のロールを割り当てます。
- Ceph Object Gateway でのユーザーの自動作成
- Keystone でのユーザーの管理
- Ceph Object Gateway は Keystone に対して、取り消されたトークンのリストを定期的にクエリーします。

6.3.3. Swift サービスの作成

Ceph Object Gateway を設定する前に、Swift サービスを有効にして Ceph Object Gateway を指定するように Keystone を設定します。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- Ceph ソフトウェアリポジトリーへのアクセス。
- OpenStack コントローラーノードへの root レベルのアクセス。

手順

- Swift サービスを作成します。

```
[root@swift~]# openstack service create --name=swift --description="Swift Service" object-store
```

このサービスを作成すると、サービス設定がエコーされます。

表6.1 例

フィールド	値
description	Swift サービス
enabled	True
id	37c4c0e79571404cb4644201a4a6e5ee
name	swift
type	object-store

6.3.4. Ceph Object Gateway エンドポイントの設定

Swift サービスを作成したら、サービスを Ceph Object Gateway に指定します。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- Ceph ソフトウェアリポジトリーへのアクセス。
- Red Hat OpenStack Platform 17.0 環境で実行中の Swift サービス。

手順

- Ceph Object Gateway を指定する OpenStack エンドポイントを作成します。

構文

```
openstack endpoint create --region REGION_NAME swift admin "URL"
openstack endpoint create --region REGION_NAME swift public "URL"
openstack endpoint create --region REGION_NAME swift internal "URL"
```

REGION_NAME は、ゲートウェイのゾーングループ名またはリージョン名に置き換えます。**URL** は、Ceph Object Gateway に適した URL に置き換えます。

例

```
[root@osp ~]# openstack endpoint create --region us-west swift admin
"http://radosgw.example.com:8080/swift/v1"
[root@osp ~]# openstack endpoint create --region us-west swift public
```

```
"http://radosgw.example.com:8080/swift/v1"
[root@osp ~]# openstack endpoint create --region us-west swift internal
"http://radosgw.example.com:8080/swift/v1"
```

フィールド	値
adminurl	**http://radosgw.example.com:8080/swift/v1**
id	**e4249d2b60e44743a67b5e5b38c18dd3**
internalurl	**http://radosgw.example.com:8080/swift/v1**
publicurl	**http://radosgw.example.com:8080/swift/v1**
region	**us-west**
service_id	**37c4c0e79571404cb4644201a4a6e5ee**
service_name	**swift**
service_type	**object-store**

エンドポイントを設定すると、サービスエンドポイントの設定が出力されます。

6.3.5. Openstack が Ceph Object Gateway エンドポイントを使用していることを確認

Swift サービスを作成し、エンドポイントを設定したら、すべての設定が正しいことを確認します。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- Ceph ソフトウェアリポジトリーへのアクセス。

手順

1. Swift サービスの下のエンドポイントをリスト表示します。

```
[root@swift~]# openstack endpoint list --service=swift
```

2. 前のコマンドでリスト表示されたエンドポイントの設定を確認します。

構文

```
[root@swift~]# openstack endpoint show ENDPOINT_ID
```

エンドポイントを表示すると、エンドポイントの設定とサービス設定が表示されます。

表6.2 例

フィールド	値
adminurl	http://radosgw.example.com:8080/swift/v1
enabled	True
id	e4249d2b60e44743a67b5e5b38c18dd3
internalurl	http://radosgw.example.com:8080/swift/v1
publicurl	http://radosgw.example.com:8080/swift/v1
region	us-west
service_id	37c4c0e79571404cb4644201a4a6e5ee
service_name	swift
service_type	object-store

関連情報

- エンドポイントの詳細を取得する方法の詳細は、Red Hat OpenStack ガイドの エンドポイントの表示 を参照してください。

6.3.6. Keystone SSL を使用するように Ceph Object Gateway を設定

Keystone が使用する OpenSSL 証明書を変換すると、Ceph Object Gateway が Keystone と連携するように設定されます。Ceph Object Gateway が OpenStack の Keystone 認証と対話すると、Keystone は自己署名 SSL 証明書で終了します。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- Ceph ソフトウェアリポジトリーへのアクセス。

手順

1. OpenSSL 証明書を **nss db** 形式に変換します。

   例

```
[root@osp ~]# mkdir /var/ceph/nss

[root@osp ~]# openssl x509 -in /etc/keystone/ssl/certs/ca.pem -pubkey | \
    certutil -d /var/ceph/nss -A -n ca -t "TCu,Cu,Tuw"
```

```
[root@osp ~]# openssl x509 -in /etc/keystone/ssl/certs/signing_cert.pem -pubkey | \
    certutil -A -d /var/ceph/nss -n signing_cert -t "P,P,P"
```

2. Ceph Object Gateway を実行しているノードに Keystone の SSL 証明書をインストールします。設定可能な **rgw_keystone_verify_ssl** の値を **false** に設定します。**rgw_keystone_verify_ssl** を **false** に設定すると、ゲートウェイは証明書の検証を試行しません。

6.3.7. Keystone 認証を使用するように Ceph Object Gateway を設定

OpenStack の Keystone 認証を使用するように Red Hat Ceph Storage を設定します。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- Ceph ソフトウェアリポジトリーへのアクセス。
- 実稼働環境への **admin** 権限がある。

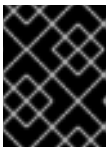
手順

1. 各ゲートウェイインスタンスで以下を行います。
   a. **nss_db_path** 設定を、NSS データベースが保存されるパスに設定します。

   例

   ```
   [ceph: root@host01 /]# ceph config set client.rgw nss_db_path
   "/var/lib/ceph/radosgw/ceph-rgw.rgw01/nss"
   ```

2. 認証証明書を指定します。
   システム管理者が OpenStack サービスを設定する方法と同様に、OpenStack Identity API 用の Keystone サービステナント、ユーザー、およびパスワードを設定することができます。ユーザー名とパスワードを指定することで、共有の秘密を **rgw_keystone_admin_token** 設定に提供するのを防ぎます。

   重要

   Red Hat は、実稼働環境で管理トークンによる認証を無効にすることを推奨します。サービステナントの認証情報には、**admin** 権限が必要です。

   必要な設定オプションは以下のとおりです。

   構文

   ```
   ceph config set client.rgw rgw_keystone_verify_ssl TRUE/FALSE
   ceph config set client.rgw rgw_s3_auth_use_keystone TRUE/FALSE
   ceph config set client.rgw rgw_keystone_api_version API_VERSION
   ceph config set client.rgw rgw_keystone_url KEYSTONE_URL:ADMIN_PORT
   ceph config set client.rgw rgw_keystone_accepted_roles ACCEPTED_ROLES_
   ceph config set client.rgw rgw_keystone_accepted_admin_roles
   ACCEPTED_ADMIN_ROLES
   ```

```
ceph config set client.rgw rgw_keystone_admin_domain default
ceph config set client.rgw rgw_keystone_admin_project SERVICE_NAME
ceph config set client.rgw rgw_keystone_admin_user KEYSTONE_TENANT_USER_NAME
ceph config set client.rgw rgw_keystone_admin_password
KEYSTONE_TENANT_USER_PASSWORD
ceph config set client.rgw rgw_keystone_implicit_tenants
KEYSTONE_IMPLICIT_TENANT_NAME
ceph config set client.rgw rgw_swift_versioning_enabled TRUE/FALSE
ceph config set client.rgw rgw_swift_enforce_content_length TRUE/FALSE
ceph config set client.rgw rgw_swift_account_in_url TRUE/FALSE
ceph config set client.rgw rgw_trust_forwarded_https TRUE/FALSE
ceph config set client.rgw rgw_max_attr_name_len
MAXIMUM_LENGTH_OF_METADATA_NAMES
ceph config set client.rgw rgw_max_attrs_num_in_req
MAXIMUM_NUMBER_OF_METADATA_ITEMS
ceph config set client.rgw rgw_max_attr_size
MAXIMUM_LENGTH_OF_METADATA_VALUE
ceph config set client.rgw rgw_keystone_accepted_reader_roles SwiftSystemReader
```

例

```
[ceph: root@host01 /]# ceph config set client.rgw rgw_keystone_verify_ssl false
[ceph: root@host01 /]# ceph config set client.rgw rgw_s3_auth_use_keystone true
[ceph: root@host01 /]# ceph config set client.rgw rgw_keystone_api_version 3
[ceph: root@host01 /]# ceph config set client.rgw rgw_keystone_url http://<public Keystone
endpoint>:5000/
[ceph: root@host01 /]# ceph config set client.rgw rgw_keystone_accepted_roles 'member,
Member, admin'
[ceph: root@host01 /]# ceph config set client.rgw rgw_keystone_accepted_admin_roles
'ResellerAdmin, swiftoperator'
[ceph: root@host01 /]# ceph config set client.rgw rgw_keystone_admin_domain default
[ceph: root@host01 /]# ceph config set client.rgw rgw_keystone_admin_project service
[ceph: root@host01 /]# ceph config set client.rgw rgw_keystone_admin_user swift
[ceph: root@host01 /]# ceph config set client.rgw rgw_keystone_admin_password password
[ceph: root@host01 /]# ceph config set client.rgw rgw_keystone_implicit_tenants true
[ceph: root@host01 /]# ceph config set client.rgw rgw_swift_versioning_enabled  true
[ceph: root@host01 /]# ceph config set client.rgw rgw_swift_enforce_content_length true
[ceph: root@host01 /]# ceph config set client.rgw rgw_swift_account_in_url true
[ceph: root@host01 /]# ceph config set client.rgw rgw_trust_forwarded_https true
[ceph: root@host01 /]# ceph config set client.rgw rgw_max_attr_name_len 128
[ceph: root@host01 /]# ceph config set client.rgw rgw_max_attrs_num_in_req 90
[ceph: root@host01 /]# ceph config set client.rgw rgw_max_attr_size  1024
[ceph: root@host01 /]# ceph config set client.rgw rgw_keystone_accepted_reader_roles
SwiftSystemReader
```

Ceph Object Gateway ユーザーは Keystone の **tenant** にマッピングされます。Keystone ユーザーには、複数のテナントで異なるロールが割り当てられている可能性があります。Ceph Object Gateway がチケットを取得する際には、テナントと、そのチケットに割り当てられたユーザーロールを確認し、設定可能な **rgw_keystone_accepted_roles** に従って要求を受け入れるか拒否します。

関連情報

- Red Hat OpenStack Platform の **ユーザーおよびアイデンティティー管理ガイド** を参照してください。

6.3.8. Ceph Object Gateway デーモンの再起動

Ceph Object Gateway を再起動すると、アクティブな設定変更を行う必要があります。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- Ceph ソフトウェアリポジトリーへのアクセス。
- 実稼働環境への **admin** 権限

手順

- Ceph 設定ファイルを保存して各 Ceph ノードに分散したら、Ceph Object Gateway インスタンスを再起動します。

注記

NAME 列の **ceph orch ps** コマンドの出力を使用して、**SERVICE_TYPE.ID** 情報を取得します。

a. ストレージクラスター内の個別のノードで Ceph Object Gateway を再起動するには、以下を実行します。

構文

```
systemctl restart ceph-CLUSTER_ID@SERVICE_TYPE.ID.service
```

例

```
[root@host01 ~]# systemctl restart ceph-c4b34c6f-8365-11ba-dc31-
529020a7702d@rgw.realm.zone.host01.gwasto.service
```

b. ストレージクラスター内のすべてのノードで Ceph Object Gateway を再起動するには、以下を実行します。

構文

```
ceph orch restart SERVICE_TYPE
```

例

```
[ceph: root@host01 /]# ceph orch restart rgw
```

第7章 セキュリティー

ストレージ管理者は、ストレージクラスター環境のセキュリティー保護に重要です。Red Hat Ceph Storage は、Ceph Object Gateway のアクセスポイントのセキュリティーを保護する暗号化および鍵管理を提供します。

前提条件

- 正常かつ実行中の Red Hat Ceph Storage クラスター
- Ceph Object Gateway ソフトウェアのインストール。

7.1. サーバー側暗号化 (SSE)

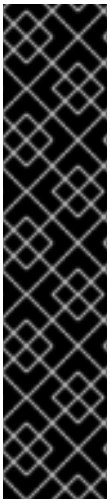
Ceph Object Gateway は、S3 アプリケーションプログラムインターフェイス (API) のアップロードされたオブジェクトのサーバー側の暗号化をサポートします。サーバー側の暗号化とは、S3 クライアントが暗号化されていない形式で HTTP 経由でデータを送信し、Ceph Object Gateway はそのデータを暗号化した形式で Red Hat Ceph Storage に保存することを意味します。

注記

- Red Hat は、Static Large Object (SLO) または Dynamic Large Object (DLO) の S3 オブジェクト暗号化をサポートしません。
- 現在、Server-Side Encryption (SSE) モードのいずれも **CopyObject** のサポートを実装していません。これは、現在開発中です。[BZ#2149450].

重要

既知の問題により、サーバー側の暗号化はマルチサイトレプリケーションと互換性がありません。この問題は将来のリリースで解決される予定です。詳細は、**既知の問題: マルチサイトオブジェクトゲートウェイ** を参照してください。



重要

暗号化を使用するには、クライアントリクエストは、SSL 接続上でリクエストを送信する **必要があります**。Red Hat は、Ceph Object Gateway が SSL を使用しない限り、クライアントからの S3 暗号化をサポートしません。ただし、テストの目的で、管理者は **ceph config set client.rgw** コマンドを使用して、**rgw_crypt_require_ssl** 設定を **false** に設定してから、Ceph Object Gateway インスタンスを再起動することで、テスト中に SSL を無効にできます。

実稼働環境では、SSL 経由で暗号化された要求を送信できない場合があります。このような場合は、サーバー側の暗号化で HTTP を使用して要求を送信します。

サーバー側の暗号化で HTTP を設定する方法は、以下の**関連情報**セクションを参照してください。

暗号化キーの管理には 3 つのオプションがあります。

お客様提供のキー

お客様が提供する鍵を使用する場合、S3 クライアントは暗号鍵を各リクエストと共に渡して、暗号化されたデータの読み取りまたは書き込みを行います。これらのキーを管理するのは、お客様の責任です。各オブジェクトの暗号化に使用する Ceph Object Gateway の鍵を覚えておく必要があります。

Ceph Object Gateway は、Amazon SSE-C 仕様に従って、S3 API で顧客提供のキー動作を実装します。

お客様がキー管理を処理し、S3 クライアントはキーを Ceph Object Gateway に渡すため、Ceph Object Gateway ではこの暗号化モードをサポートするための特別な設定は必要ありません。

キー管理サービス

キー管理サービスを使用する場合、セキュアなキー管理サービスはキーを格納し、Ceph Object Gateway はデータの暗号化または復号の要求に対応するためにキーをオンデマンドで取得します。

Ceph Object Gateway は、Amazon SSE-KMS 仕様に従って S3 API にキー管理サービスの動作を実装します。

重要

現時点で、テスト済み鍵管理の実装は HashiCorp Vault および OpenStack Barbican です。ただし、OpenStack Barbican はテクノロジープレビューであるため、実稼働システムでの使用はサポートされません。

SSE-S3

SSE-S3 を使用する場合、キーはボールトに保存されますが、Ceph によって自動的に作成および削除され、データの暗号化または復号化の要求を処理するために必要に応じて取得されます。

Ceph Object Gateway は、Amazon SSE-S3 仕様に従って S3 API に SSE-S3 動作を実装します。

関連情報

- Amazon SSE-C
- Amazon SSE-KMS
- **サーバー側の暗号化の設定**
- **HashiCorp Vault**

7.1.1. 既存 S3 バケットのデフォルトの暗号化設定

ストレージ管理者は、既存の Amazon S3 バケットにデフォルトの暗号化を設定して、すべてのオブジェクトがバケットに保存されるときに暗号化されるようにできます。バケット暗号化 API を使用して、Amazon S3 で管理されたキー (SSE-S3) または Amazon KMS カスタマーマスターキー (SSE-KMS) を使用したサーバー側での暗号化をサポートできます。

注記

SSE-KMS は Red Hat Ceph Storage 5.x 以降でのみサポートされ、Red Hat Ceph Storage 4.x ではサポートされません。

PutBucketEncryption API を使用して、既存 Amazon S3 バケットのデフォルトの暗号化を管理できます。このバケットにアップロードされたすべてのファイルには、バケットレベルでデフォルトの暗号化を定義することにより、この暗号化が適用されます。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- Ceph Object Gateway のインストール
- S3 バケットが作成されている。
- ユーザーアクセスで作成された S3 ユーザー。
- AWS CLI パッケージがインストールされた Ceph Object Gateway クライアントへのアクセス。

手順

1. 暗号化設定用の JSON ファイルを作成します。

   例

   ```
   [user@client ~]$ vi bucket-encryption.json
   ```

2. 暗号化設定ルールをファイルに追加します。

   例

   ```
   {
    "Rules": [
          {
       "ApplyServerSideEncryptionByDefault": {
                     "SSEAlgorithm": "AES256"
       }
         }
      ]
   }
   ```

3. バケットのデフォルトの暗号化を設定します。

   構文

   ```
   aws --endpoint-url=pass:q[_RADOSGW_ENDPOINT_URL_]:pass:q[_PORT_] s3api put-bucket-encryption --bucket pass:q[_BUCKET_NAME_] --server-side-encryption-configuration pass:q[_file://PATH_TO_BUCKET_ENCRYPTION_CONFIGURATION_FILE/BUCKET_ENCRYPTION_CONFIGURATION_FILE.json_]
   ```

   例

   ```
   [user@client ~]$ aws --endpoint-url=http://host01:80 s3api put-bucket-encryption --bucket testbucket --server-side-encryption-configuration file://bucket-encryption.json
   ```

検証

- バケットのバケット暗号化設定を取得します。

  構文

  ```
  aws --endpoint-url=pass:q[_RADOSGW_ENDPOINT_URL_]:pass:q[_PORT_] s3api get-bucket-encryption --bucket BUCKET_NAME
  ```

  例

  ```
  [user@client ~]$  aws --profile ceph --endpoint=http://host01:80 s3api get-bucket-encryption --bucket testbucket

  {
      "ServerSideEncryptionConfiguration": {
          "Rules": [
              {
                  "ApplyServerSideEncryptionByDefault": {
                      "SSEAlgorithm": "AES256"
                  }
              }
          ]
      }
  }
  ```

注記

バケットにデフォルトの暗号化設定がない場合、**get-bucket-encryption** コマンドは **ServerSideEncryptionConfigurationNotFoundError** を返します。

7.1.2. デフォルトのバケット暗号化の削除

s3api delete-bucket-encryption コマンドを使用して、指定したバケットのデフォルトのバケット暗号化を削除できます。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- Ceph Object Gateway のインストール
- S3 バケットが作成されている。
- ユーザーアクセスで作成された S3 ユーザー。
- AWS CLI パッケージがインストールされた Ceph Object Gateway クライアントへのアクセス。

手順

- バケット暗号化設定を削除します。

  構文

  ```
  aws --endpoint-url=RADOSGW_ENDPOINT_URL:PORT s3api delete-bucket-encryption --bucket BUCKET_NAME
  ```

-

例

```
[user@client ~]$ aws --endpoint-url=http://host01:80 s3api delete-bucket-encryption --bucket
testbucket
```

検証

- バケットのバケット暗号化設定を取得します。

  構文

  ```
  aws --endpoint-url=RADOSGW_ENDPOINT_URL:PORT s3api get-bucket-encryption --
  bucket BUCKET_NAME
  ```

  例

  ```
  [user@client ~]$  aws --endpoint=http://host01:80 s3api get-bucket-encryption --bucket
  testbucket

  An error occurred (ServerSideEncryptionConfigurationNotFoundError) when calling the
  GetBucketEncryption operation:
  The server side encryption configuration was not found
  ```

7.2. サーバー側の暗号化要求

実稼働環境では、クライアントはプロキシーを介して Ceph Object Gateway に接続されることがよくあります。このプロキシーは、複数の Ceph Object Gateway に接続するため、ロードバランサーと呼ばれます。クライアントが Ceph Object Gateway に要求を送信する場合、ロードバランサーはこれらの要求を複数の Ceph Object Gateway にルーティングし、ワークロードを配布します。

このような設定では、ロードバランサーでも、ロードバランサーと複数の Ceph Object Gateways の間でも、SSL の終了が発生する可能性があります。通信は HTTP のみを使用して行われます。サーバー側の暗号化要求を受け入れるように Ceph Object Gateway を設定するには、**サーバー側の暗号化の設定**を参照してください。

7.3. サーバー側の暗号化の設定

SSL 経由で暗号化された要求を送信できない場合に、HTTP を使用して Ceph Object Gateway に要求を送信するようにサーバー側の暗号化を設定できます。

以下の手順では、HAProxy をプロキシーおよびロードバランサーとして使用します。

前提条件

- 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。
- 稼働中の Red Hat Ceph Storage クラスターがある。
- Ceph Object Gateway ソフトウェアのインストール。
- HAProxy ソフトウェアのインストール。

手順

1. **haproxy.cfg** ファイルを編集します。

   例

```
frontend http_web *:80
    mode http
    default_backend rgw

frontend rgw-https
  bind *:443 ssl crt /etc/ssl/private/example.com.pem
  default_backend rgw

backend rgw
    balance roundrobin
    mode http
    server  rgw1 10.0.0.71:8080 check
    server  rgw2 10.0.0.80:8080 check
```

2. **http** フロントエンドへのアクセスを許可する行をコメントアウトし、代わりに **https** フロントエンドを使用するように HAProxy に指示する手順を追加します。

   例

```
#     frontend http_web *:80
#     mode http
#     default_backend rgw

frontend rgw-https
  bind *:443 ssl crt /etc/ssl/private/example.com.pem
  http-request set-header X-Forwarded-Proto https if { ssl_fc }
  http-request set-header X-Forwarded-Proto https
# here we set the incoming HTTPS port on the load balancer (eg : 443)
  http-request set-header X-Forwarded-Port 443
  default_backend rgw

backend rgw
    balance roundrobin
    mode http
    server  rgw1 10.0.0.71:8080 check
    server  rgw2 10.0.0.80:8080 check
```

3. **rgw_trust_forwarded_https** オプションを **true** に設定します。

   例

```
[ceph: root@host01 /]# ceph config set client.rgw rgw_trust_forwarded_https true
```

4. HAProxy を有効にして起動します。

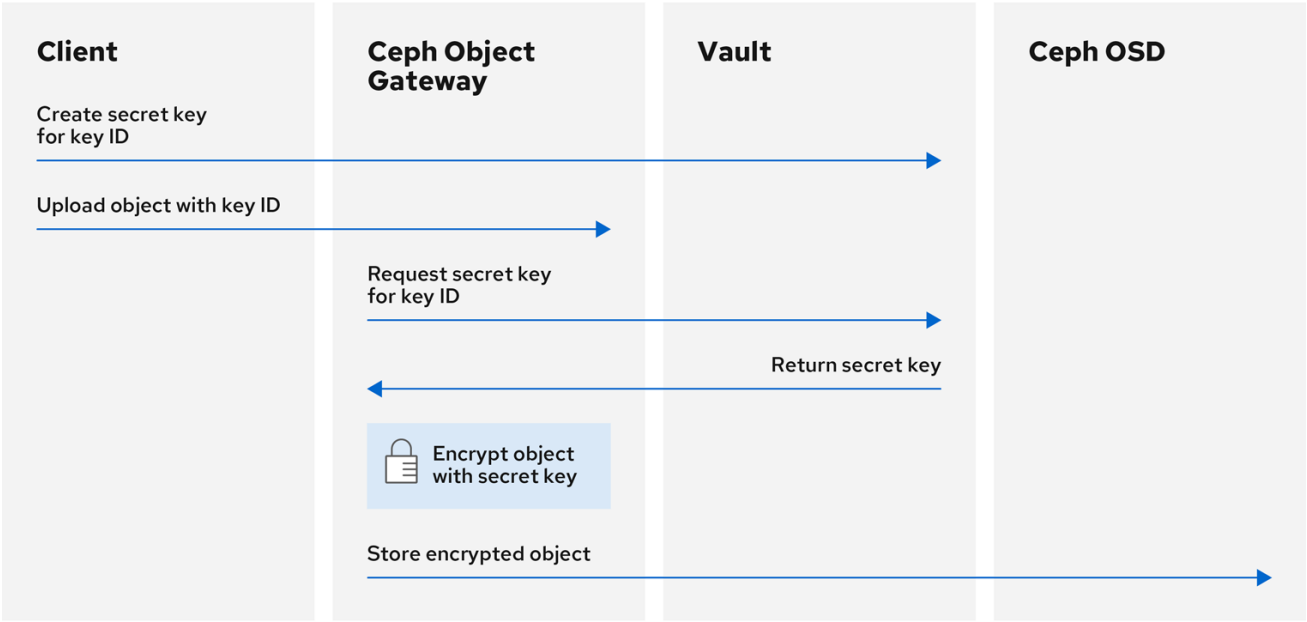
```
[root@host01 ~]# systemctl enable haproxy
[root@host01 ~]# systemctl start haproxy
```

関連情報

- 詳細は、**Red Hat Ceph Storage Object Gateway ガイド** の 高可用性サービス セクションを参照してください。
- 詳細は、**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 の章を参照してください。

7.4. HASHICORP VAULT

ストレージ管理者は、Ceph Object Gateway で使用するために、キー、パスワード、および証明書を HAshiCorp Vault に安全に保存できます。HashiCorp Vault は、Ceph Object Gateway で使用されるサーバー側の暗号化にセキュアなキー管理サービスを提供します。



基本的なワークフロー:

1. クライアントは、オブジェクトのキー ID に基づいて Vault からシークレットキーの作成を要求します。
2. クライアントはオブジェクトのキー ID を持つオブジェクトを Ceph Object Gateway にアップロードします。
3. 次に、Ceph Object Gateway は Vault から新規に作成されたシークレットキーを要求します。
4. Vault は、Ceph Object Gateway に秘密鍵を返して要求に応えます。
5. これで、Ceph Object Gateway は、新しい秘密鍵を使用してオブジェクトを暗号化できます。
6. 暗号化が完了すると、オブジェクトが Ceph OSD に保存されます。

重要

Red Hat は、弊社のテクノロジーパートナーと協力して、本書をお客様にサービスとして提供します。ただし、Red Hat はこの製品のサポートを提供しません。この製品の技術的なサポートが必要な場合は、Hashicorp にサポートを依頼してください。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- Ceph Object Gateway ソフトウェアのインストール。
- HashiCorp Vault ソフトウェアのインストール。

7.4.1. Vault のシークレットエンジン

HashiCorp Vault は、データを生成、保存、または暗号化するためのシークレットエンジンを複数提供します。アプリケーションプログラミングインターフェイス (API) は、データに対するアクションを求めるデータ呼び出しをシークレットエンジンに送信し、シークレットエンジンはそのアクション要求の結果を返します。

Ceph Object Gateway は、HashiCorp Vault シークレットエンジンの 2 つをサポートします。

- キー/値のバージョン 2
- Transit

キー/値のバージョン 2

Key/Value シークレットエンジンは、ディスク上の Vault 内にランダムのシークレットを保存します。**kv** エンジンのバージョン 2 では、設定可能な数のバージョン数をキーに指定できます。デフォルトのバージョン数は 10 です。バージョンを削除しても元のデータは削除されませんが、データが削除されたことになり、削除されたバージョンを元に戻すことができます。API エンドポイントまたは **destroy** コマンドを使用して、バージョンのデータを完全に削除できます。キーのバージョンおよびメタデータをすべて削除するには、**metadata** コマンドまたは API エンドポイントを使用することができます。鍵名は文字列にする必要があります。また、コマンドラインインターフェイスの使用時に、エンジンは文字列以外の値を文字列に変換します。文字列以外の値を保持するには、JSON ファイルを指定するか、HTTP アプリケーションプログラミングインターフェイス (API) を使用します。

注記

アクセス制御リスト (ACL) ポリシーの場合、キー/値のシークレットエンジンは **作成** 機能と **更新** 機能の違いを認識します。

Transit

Transit シークレットエンジンは、移動データで暗号化機能を実行します。Transit シークレットエンジンはハッシュを生成したり、ランダムバイトのソースとなったり、データの署名や検証を行うことができます。Vault は、Transit シークレットエンジンの使用時にデータを保存しません。Transit シークレットエンジンは、複数の目的で同じ鍵を使用できるようにすることで鍵導出をサポートします。また、transit シークレットエンジンは鍵バージョン管理をサポートします。Transit シークレットエンジンは、以下のキータイプをサポートします。

aes128-gcm96

128 ビットの AES キーと 96 ビットのノンスを備えた AES-GCM。暗号化、復号、鍵導出、および収束暗号化をサポートします。

aes256-gcm96

256 ビットの AES キーと 96 ビットのノンスを備えた AES-GCM。暗号化、復号、鍵導出、および収束暗号化 (デフォルト) をサポートします。

chacha20-poly1305

256 ビットキーの ChaCha20-Poly1305。暗号化、復号、鍵導出、および収束暗号化をサポートします。

ed25519

Ed25519。署名、署名の検証、および鍵導出をサポートします。

ecdsa-p256

曲線 P-256 を使用した ECDSA。署名および署名の検証をサポートします。

ecdsa-p384

曲線 P-384 を使用した ECDSA。署名および署名の検証をサポートします。

ecdsa-p521

曲線 P-521 を使用した ECDSA。署名および署名の検証をサポートします。

rsa-2048

2048 ビット RSA 鍵、暗号化、復号、署名、および署名の検証をサポートします。

rsa-3072

3072 ビット RSA 鍵。暗号化、復号、署名、および署名の検証をサポートします。

rsa-4096

4096 ビットの RSA 鍵。暗号化、復号、署名、および署名の検証をサポートします。

関連情報

- 詳細は、Vault のプロジェクトサイトにある KV Secrets Engine ドキュメントを参照してください。
- 詳細は、Vault のプロジェクトサイトにある Transport Secrets Engine ドキュメントを参照してください。

7.4.2. Vault の認証

HashiCorp Vault は、複数のタイプの認証メカニズムをサポートします。Ceph Object Gateway は現在 Vault エージェントメソッドをサポートしています。Ceph Object Gateway は **rgw_crypt_vault_auth** オプションおよび **rgw_crypt_vault_addr** オプションを使用して HashiCorp Vault を使用するように設定します。

重要

Red Hat は、コンテナーの認証方法として Vault Agent の使用をサポートしており、トークン認証の使用は、コンテナーではサポートされていません。

Vault エージェント

Vault エージェントは、クライアントノードで実行するデーモンで、トークンの更新と共にクライアント側のキャッシュを提供します。Vault エージェントは、通常 Ceph Object Gateway ノードで実行されます。Vault エージェントを実行し、トークンファイルを更新します。Vault エージェントをこのモードで使用すると、ファイルシステムのパーミッションを使用して、トークンの使用にアクセスできるユーザーを制限できます。また、Vault エージェントはプロキシーサーバーとしても機能します。つまり、Vault は必要に応じてトークンを追加し、渡された要求にトークンを追加してから実際のサーバーに転送します。Vault エージェントは、トークンをファイルシステムに格納する際にもトークンの更新を処理できます。たとえば、Vault エージェントがローカルホストのみをリッスンするなど、Ceph Object Gateway が Vault エージェントとの接続に使用するネットワークのセキュリティーを保護する必要があります。

関連情報

関連情報

- 詳細は、Vault のプロジェクトサイトにある Vault Agent ドキュメントを参照してください。

7.4.3. Vault の namespace

HashiCorp Vault をエンタープライズサービスとして使用すると、組織内のチームが使用可能な分離された名前空間の一元管理が提供されます。これらの分離された名前空間環境は **テナント** と呼ばれ、組織内のチームがこれらの **テナント** を使用して、ポリシー、シークレット、ID を他のチームから分離することができます。Vault の名前空間機能は、単一インフラストラクチャー内からセキュアなマルチテナンシーをサポートします。

関連情報

- 詳細は、Vault のプロジェクトサイトにある Vault Enterprise Namespaces ドキュメントを参照してください。

7.4.4. 永続エンジンの互換性サポート

以前のバージョンの Ceph は、簡単なキーストアとして Transit エンジンを使用した場合の互換性サポートがあります。Transit エンジンの **compat** オプションを使用して、互換性サポートを設定できます。以下のコマンドを使用して、以前のサポートを無効にすることができます。

例

```
[ceph: root@host03 /]# ceph config set client.rgw rgw_crypt_vault_secret_engine transit compat=0
```

注記

これは今後のバージョンでデフォルトであり、新規インストールに現行バージョンを使用できます。

現在のバージョンにおける通常のデフォルトは以下のとおりです。

例

```
[ceph: root@host03 /]# ceph config set client.rgw rgw_crypt_vault_secret_engine transit compat=1
```

これにより、新しく作成されたオブジェクトの新規エンジンが有効になり、古いオブジェクトに古いエンジンを使用できるようになります。古いオブジェクトおよび新しいオブジェクトにアクセスするには、Vault トークンに以前の転送ポリシーと新しい転送ポリシーの両方が必要です。

以下のコマンドを使用すると、古いエンジンのみが強制的に使用できます。

例

```
[ceph: root@host03 /]# ceph config set client.rgw rgw_crypt_vault_secret_engine transit compat=2
```

Vault が **export/encryption-key** で終わる場合は、このモードがデフォルトで選択されます。

重要

client.rgw オプションを設定したら、新しい値を有効にするために Ceph Object Gateway デーモンを再起動する必要があります。

関連情報

- 詳細は、Vault のプロジェクトサイトにある Vault Agent ドキュメントを参照してください。

7.4.5. Vault のトークンポリシーの作成

トークンポリシーは、全 Vault トークンが持つ権限を指定します。1 つのトークンに複数のポリシーを持たせることができます。設定に必要なポリシーを使用する必要があります。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- HashiCorp Vault ソフトウェアのインストール。
- HashiCorp Vault ノードへの root レベルのアクセス。

手順

1. トークンポリシーを作成します。
   a. キー/値シークレットエンジンの場合:

      例

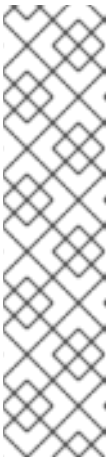
      ```
      [root@vault ~]# vault policy write rgw-kv-policy -<<EOF
       path "secret/data/*" {
         capabilities = ["read"]
       }
      EOF
      ```

   b. Transit エンジンの場合:

      例

      ```
      [root@vault ~]# vault policy write rgw-transit-policy -<<EOF
        path "transit/keys/*" {
          capabilities = [ "create", "update" ]
          denied_parameters = {"exportable" = [], "allow_plaintext_backup" = [] }
        }

        path "transit/keys/*" {
          capabilities = ["read", "delete"]
        }

        path "transit/keys/" {
          capabilities = ["list"]
        }

        path "transit/keys/+/rotate" {
      ```

```
  capabilities = [ "update" ]
 }

 path "transit/*" {
  capabilities = [ "update" ]
 }
EOF
```



注記

以前のバージョンの Ceph で Transit シークレットエンジンを使用していた場合、トークンポリシーは以下のようになります。

例

```
[root@vault ~]# vault policy write old-rgw-transit-policy -<<EOF
  path "transit/export/encryption-key/*" {
    capabilities = ["read"]
  }
EOF
```

SSE-KMS と SSE-S3 の両方を使用している場合は、それぞれ別のコンテナーを指す必要があります。個別の Vault インスタンスを使用するか、共通の中継点の下に個別に中継インスタンスまたは別のブランチをマウントすることができます。個別の Vault インスタンスを使用していない場合は、**rgw_crypt_vault_prefix** および **rgw_crypt_sse_s3_vault_prefix** を使用して、SSE-KMS または SSE-S3 を指定してコンテナーを分離することができます。SSE-KMS バケット所有者に Vault 権限を付与する場合、SSE-S3 キーへの権限を付与しないでください。Ceph のみが SSE-S3 キーにアクセスできる必要があります。

7.4.6. Vault で SSE-KMS を使用するための Ceph Object Gateway の設定

キー管理に SSE-KMS で HashiCorp Vault を使用するように Ceph Object Gateway を設定するには、それを暗号化キーストアとして設定する必要があります。現在、Ceph Object Gateway は 2 つの異なるシークレットエンジンと、2 つの異なる認証方法をサポートしています。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- Ceph Object Gateway ソフトウェアのインストール。
- Ceph Object Gateway ノードへのルートレベルのアクセス。

手順

1. **ceph config set client.rgw OPTION VALUE** コマンドを使用して、Vault を暗号化キーストアとして有効にします。

   構文

   ```
   ceph config set client.rgw rgw_crypt_s3_kms_backend vault
   ```

2. 以下のオプションおよび値を追加します。

構文

```
ceph config set client.rgw rgw_crypt_vault_auth agent
ceph config set client.rgw rgw_crypt_vault_addr http://VAULT_SERVER:8100
```

3. ユースケースに従ってポリシーをカスタマイズします。

4. role-id を取得します。

構文

```
vault read auth/approle/role/rgw-ap/role-id -format=json | \ jq -r .data.role_id >
PATH_TO_FILE
```

5. secret-id を取得します。

構文

```
vault read auth/approle/role/rgw-ap/role-id -format=json | \ jq -r .data.secret_id >
PATH_TO_FILE
```

6. Vault エージェントの設定を作成します。

例

```
pid_file = "/run/kv-vault-agent-pid"
auto_auth {
  method "AppRole" {
    mount_path = "auth/approle"
    config = {
      role_id_file_path ="/root/vault_configs/kv-agent-role-id"
      secret_id_file_path ="/root/vault_configs/kv-agent-secret-id"
      remove_secret_id_file_after_reading ="false"
    }
  }
}
cache {
  use_auto_auth_token = true
}
listener "tcp" {
  address = "127.0.0.1:8100"
  tls_disable = true
}
vault {
  address = "http://10.8.128.9:8200"
}
```

7. systemctl を使用して永続デーモンを実行します。

例

```
[root@host03 ~]# /usr/local/bin/vault agent -config=/usr/local/etc/vault/rgw-agent.hcl
```

8. Vault エージェントの実行時に、トークンファイルに有効なトークンが設定されます。
9. Vault シークレットエンジン (キー/値または Transit) を選択します。
   a. **Key/Value** を使用している場合は、以下の行を追加します。

      例

      ```
      [ceph: root@host03 /]# ceph config set client.rgw rgw_crypt_vault_secret_engine kv
      ```

   b. **Transit** を使用している場合は、以下の行を追加します。

      例

      ```
      [ceph: root@host03 /]# ceph config set client.rgw rgw_crypt_vault_secret_engine transit
      ```

10. **ceph config set client.rgw OPTION VALUE** コマンドを使用して、Vault 名前空間が暗号化の鍵を取得するように設定します。

    例

    ```
    [ceph: root@host03 /]# ceph config set client.rgw rgw_crypt_vault_namespace
    testnamespace1
    ```

11. パス接頭辞を設定し、Ceph Object Gateway が Vault から暗号化キーを取得する場所を制限します。

    例

    ```
    [ceph: root@host03 /]# ceph config set client.rgw rgw_crypt_vault_prefix /v1/secret/data
    ```

    a. エクスポート可能な Transit キーの場合、以下のように接頭辞パスを設定します。

       例

       ```
       [ceph: root@host03 /]# ceph config set client.rgw rgw_crypt_vault_prefix
       /v1/transit/export/encryption-key
       ```

       Vault サーバーのドメイン名が **vault-server** である場合は、Ceph Object Gateway は以下の URL から暗号化されたトランジションキーを取得します。

       例

       ```
       http://vault-server:8200/v1/transit/export/encryption-key
       ```

12. Ceph Object Gateway デーモンを再起動します。
    a. ストレージクラスター内の個別のノードで Ceph Object Gateway を再起動するには、以下を実行します。

       構文

       ```
       systemctl restart ceph-CLUSTER_ID@SERVICE_TYPE.ID.service
       ```

例

```
[root@host03 ~]# systemctl restart ceph-c4b34c6f-8365-11ba-dc31-
529020a7702d@rgw.realm.zone.host01.gwasto.service
```

b. ストレージクラスター内のすべてのノードで Ceph Object Gateway を再起動するには、以下を実行します。

構文

```
ceph orch restart SERVICE_TYPE
```

例

```
[ceph: root@host03 /]# ceph orch restart rgw
```

関連情報

- 詳細は、**Red Hat Ceph Storage Object Gateway ガイド**の **Vault のシークレットエンジン** セクションを参照してください。
- 詳細は、**Red Hat Ceph Storage Object Gateway ガイド**の **Vault の認証** セクションを参照してください。

7.4.7. Vault で SSE-S3 を使用するための Ceph Object Gateway の設定

キー管理に SSE-S3 で HashiCorp Vault を使用するように Ceph Object Gateway を設定するには、それを暗号化キーストアとして設定する必要があります。現在 Ceph Object Gateway が使用している認証方法は **agent** だけです。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- Ceph Object Gateway ソフトウェアのインストール。
- Ceph Object Gateway ノードへのルートレベルのアクセス。

手順

1. Cephadm シェルにログインします。

   例

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

2. SSE-S3 暗号化キーを取得するシークレットエンジンとして Vault を有効にします。

   構文

   ```
   ceph config set client.rgw rgw_crypt_sse_s3_backend vault
   ```

3. SSE-S3 および Vault で使用する認証方法を設定するには、以下を設定します。

構文

```
ceph config set client.rgw rgw_crypt_sse_s3_vault_auth agent
ceph config set client.rgw rgw_crypt_sse_s3_vault_addr
http://VAULT_AGENT:VAULT_AGENT_PORT
```

例

```
[ceph: root@host01 ~]# ceph config set client.rgw rgw_crypt_sse_s3_vault_auth agent
[ceph: root@host01 ~]# ceph config set client.rgw rgw_crypt_sse_s3_vault_addr
http://vaultagent:8100
```

a. ユースケースに応じてポリシーをカスタマイズして、Vault エージェントを設定します。

b. role-id を取得します。

構文

```
vault read auth/approle/role/rgw-ap/role-id -format=json | \ jq -r .rgw-ap-role-id >
PATH_TO_FILE
```

c. secret-id を取得します。

構文

```
vault read auth/approle/role/rgw-ap/role-id -format=json | \ jq -r .rgw-ap-secret-id >
PATH_TO_FILE
```

d. Vault エージェントの設定を作成します。

例

```
pid_file = "/run/rgw-vault-agent-pid"
auto_auth {
  method "AppRole" {
    mount_path = "auth/approle"
    config = {
      role_id_file_path ="/usr/local/etc/vault/.rgw-ap-role-id"
      secret_id_file_path ="/usr/local/etc/vault/.rgw-ap-secret-id"
      remove_secret_id_file_after_reading ="false"
    }
  }
}
cache {
  use_auto_auth_token = true
}
listener "tcp" {
  address = "127.0.0.1:8100"
  tls_disable = true
}
```

```
vault {
  address = "https://vaultserver:8200"
}
```

e. systemctl を使用して永続デーモンを実行します。

例

```
[root@host01 ~]# /usr/local/bin/vault agent -config=/usr/local/etc/vault/rgw-agent.hcl
```

f. Vault エージェントの実行時に、トークンファイルに有効なトークンが設定されます。

4. 暗号化キー (キー/値またはトランジット) の取得に使用する Vault シークレットエンジンを設定します。

a. **Key/Value** を使用する場合は、次のように設定します。

例

```
[ceph: root@host01 /]# ceph config set client.rgw rgw_crypt_sse_s3_vault_secret_engine
kv
```

b. **Transit** を使用する場合は、次のように設定します。

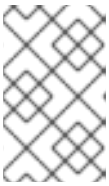
例

```
[ceph: root@host01 /]# ceph config set client.rgw rgw_crypt_sse_s3_vault_secret_engine
transit
```

5. オプション: Ceph Object Gateway が特定の名前空間内の Vault にアクセスして暗号化キーを取得するように設定します。

例

```
[ceph: root@host01 /]# ceph config set client.rgw rgw_crypt_sse_s3_vault_namespace
company/testnamespace1
```



注記

Vault 名前空間を使用すると、チームはテナントと呼ばれる隔離された環境内で運用できます。Vault 名前空間機能は、Vault Enterprise バージョンでのみ使用できます。

6. オプション: Ceph Object Gateway が暗号鍵を取得する URL パス接頭辞を設定して、Vault シークレットスペースの特定のサブセットへのアクセスを制限します。

a. **Key/Value** を使用する場合は、次のように設定します。

例

```
[ceph: root@host01 /]# ceph config set client.rgw rgw_crypt_sse_s3_vault_prefix
/v1/secret/data
```

b. **Transit** を使用する場合は、次のように設定します。

例

```
[ceph: root@host01 /]# ceph config set client.rgw rgw_crypt_sse_s3_vault_prefix
/v1/transit
```

Vault サーバーのドメイン名が **vault-server** である場合は、Ceph Object Gateway は以下の URL から暗号化されたトランジションキーを取得します。

例

```
http://vaultserver:8200/v1/transit
```

7. オプション: カスタム SSL 証明書を使用して Vault で認証するには、次の設定を設定します。

構文

```
ceph config set client.rgw rgw_crypt_sse_s3_vault_verify_ssl true
ceph config set client.rgw rgw_crypt_sse_s3_vault_ssl_cacert
PATH_TO_CA_CERTIFICATE
ceph config set client.rgw rgw_crypt_sse_s3_vault_ssl_clientcert
PATH_TO_CLIENT_CERTIFICATE
ceph config set client.rgw rgw_crypt_sse_s3_vault_ssl_clientkey PATH_TO_PRIVATE_KEY
```

例

```
[ceph: root@host01 /]# ceph config set client.rgw rgw_crypt_sse_s3_vault_verify_ssl true
[ceph: root@host01 /]# ceph config set client.rgw rgw_crypt_sse_s3_vault_ssl_cacert
/etc/ceph/vault.ca
[ceph: root@host01 /]# ceph config set client.rgw rgw_crypt_sse_s3_vault_ssl_clientcert
/etc/ceph/vault.crt
[ceph: root@host03 /]# ceph config set client.rgw rgw_crypt_sse_s3_vault_ssl_clientkey
/etc/ceph/vault.key
```

8. Ceph Object Gateway デーモンを再起動します。

a. ストレージクラスター内の個別のノードで Ceph Object Gateway を再起動するには、以下を実行します。

構文

```
systemctl restart ceph-CLUSTER_ID@SERVICE_TYPE.ID.service
```

例

```
[root@host01 ~]# systemctl restart ceph-c4b34c6f-8365-11ba-dc31-
529020a7702d@rgw.realm.zone.host01.gwasto.service
```

b. ストレージクラスター内のすべてのノードで Ceph Object Gateway を再起動するには、以下を実行します。

構文

```
ceph orch restart SERVICE_TYPE
```

例

```
[ceph: root@host01 /]# ceph orch restart rgw
```

関連情報

- 詳細は、**Red Hat Ceph Storage Object Gateway ガイド**の Vault のシークレットエンジン セクションを参照してください。
- 詳細は、**Red Hat Ceph Storage Object Gateway ガイド**の Vault の認証 セクションを参照してください。

7.4.8. kv エンジンを使用したキーの作成

HashiCorp Vault Key/Value シークレットエンジン (**kv**) を設定し、Ceph Object Gateway で使用するためのキーを作成できるようにします。シークレットは、**kv** シークレットエンジンのキーと値のペアとして保存されます。

重要

サーバー側の暗号化のキーは 256 ビットの長さで、**base64** を使用してエンコードする必要があります。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- HashiCorp Vault ソフトウェアのインストール。
- HashiCorp Vault ノードへの root レベルのアクセス。

手順

1. キー/値 バージョン 2 シークレットエンジンを有効にします。

   例

   ```
   vault secrets enable -path secret kv-v2
   ```

2. 新しいキーを作成します。

   構文

   ```
   vault kv put secret/PROJECT_NAME/BUCKET_NAME key=$(openssl rand -base64 32)
   ```

   例

   ```
   [root@vault ~]# vault kv put secret/myproject/mybucketkey key=$(openssl rand -base64 32)

   ====== Metadata ======
   ```

```
Key              Value
---              -----
created_time     2020-02-21T17:01:09.095824999Z
deletion_time    n/a
destroyed        false
version          1
```

7.4.9. 転送エンジンを使用した鍵の作成

HashiCorp Vault 遷移シークレットエンジン (**transit**) を設定して、Ceph Object Gateway で使用するキーを作成できるようにします。Ceph Object Gateway でサーバー側の暗号化に使用するためには、Transit シークレットエンジンで作成した鍵がエクスポート可能である必要があります。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- HashiCorp Vault ソフトウェアのインストール。
- HashiCorp Vault ノードへの root レベルのアクセス。

手順

1. Transit シークレットエンジンを有効にします。

```
[root@vault ~]# vault secrets enable transit
```

2. 新しいエクスポート可能なキーを作成します。

構文

```
vault write -f transit/keys/BUCKET_NAME exportable=true
```

例

```
[root@vault ~]# vault write -f transit/keys/mybucketkey exportable=true
```

注記

デフォルトでは、上記のコマンドは **aes256-gcm96** タイプキーを作成します。

3. キーの作成を確認します。

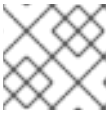
構文

```
vault read transit/export/encryption-key/BUCKET_NAME/VERSION_NUMBER
```

例

```
[root@vault ~]# vault read transit/export/encryption-key/mybucketkey/1

Key     Value
```

```
---     -----
keys    map[1:-gbTI9lNpqv/V/2lDcmH2Nq1xKn6FPDWarCmFM2aNsQ=]
name    mybucketkey
type    aes256-gcm96
```



注記

キーバージョンを含む完全なキーパスを指定する必要があります。

7.4.10. AWS および Vault を使用したオブジェクトのアップロード

Ceph Object Gateway にオブジェクトをアップロードする際に、Ceph Object Gateway は Vault からキーを取得し、そのオブジェクトをバケットに暗号化して保存します。オブジェクトのダウンロード要求が行われると、Ceph Object Gateway は Vault から対応するキーを自動的に取得し、オブジェクトを復号します。オブジェクトをアップロードするには、Ceph Object Gateway は Vault からキーを取得した後、オブジェクトを暗号化してバケットに保存します。Ceph Object Gateway は、Vault から対応するキーを取得し、オブジェクトをダウンロードする要求がある場合にオブジェクトを復号します。

注記

URL は、**rgw_crypt_vault_addr** オプションと、**rgw_crypt_vault_prefix** オプションで設定されたパス接頭辞を使用して構築されます。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- Ceph Object Gateway ソフトウェアのインストール。
- HashiCorp Vault ソフトウェアのインストール。
- Ceph Object Gateway クライアントノードへのアクセス。
- Amazon Web Services (AWS) へのアクセス。

手順

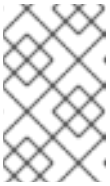
1. AWS コマンドラインクライアントを使用してオブジェクトをアップロードし、要求に Secure Side Encryption (SSE) キー ID を提供します。
   a. キー/値シークレットエンジンの場合:

例 (SSE-KMS を使用)

```
[user@client ~]$ aws --endpoint=http://radosgw:8000 s3 cp plaintext.txt
s3://mybucket/encrypted.txt --sse=aws:kms --sse-kms-key-id myproject/mybucketkey
```

例 (SSE-S3 を使用)

```
[user@client ~]$ aws s3api --endpoint http://rgw_host:8080 put-object --bucket my-
bucket --key obj1 --body test_file_to_upload --server-side-encryption AES256
```



注記

この例では、Ceph Object Gateway は **http://vault-server:8200/v1/secret/data/myproject/mybucketkey** からシークレットをフェッチします。

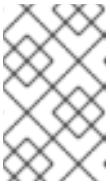
b. Transit エンジンの場合:

例 (SSE-KMS を使用)

```
[user@client ~]$ aws --endpoint=http://radosgw:8000 s3 cp plaintext.txt
s3://mybucket/encrypted.txt --sse=aws:kms --sse-kms-key-id mybucketkey
```

例 (SSE-S3 を使用)

```
[user@client ~]$ aws s3api --endpoint http://rgw_host:8080 put-object --bucket my-
bucket --key obj1 --body test_file_to_upload --server-side-encryption AES256
```



注記

この例では、Ceph Object Gateway は **http://vaultserver:8200/v1/transit/mybucketkey** からシークレットをフェッチします。

関連情報

- 詳細は、Vault のプロジェクトサイトにある Install Vault ドキュメントを参照してください。

7.5. CEPH OBJECT GATEWAY およびマルチファクター認証

ストレージ管理者は、Ceph Object Gateway ユーザーの時間ベースのワンタイムパスワード (TOTP) トークンを管理できます。

7.5.1. マルチファクター認証

バケットにオブジェクトのバージョニングを設定した場合、開発者は、必要に応じて、削除要求にマルチファクター認証 (MFA) を要求するように設定することができます。MFA を使用すると、時間ベースのワンタイムパスワード (TOTP) トークンは鍵として **x-amz-mfa** ヘッダーに渡されます。トークンは、Google Authenticator などの仮想 MFA デバイス、または Gemalto が提供するようなハードウェア MFA デバイスで生成されます。

radosgw-admin を使用して、時間ベースのワンタイムパスワードトークンをユーザーに割り当てます。シークレットシードおよびシリアル ID を設定する必要があります。**radosgw-admin** を使用して、トークンのリスト表示、削除、および再同期を行うこともできます。

重要

マルチサイト環境では、ゾーンごとに異なるトークンを使用することが推奨されます。これは、MFA ID がユーザーのメタデータに設定されている一方で、実際の MFA ワンタイムパスワード設定はローカルゾーンの OSD に存在するためです。

表7.1 用語

用語	説明
TOTP	時間ベースのワンタイムパスワード
トークンのシリアル	TOTP トークンの ID を表す文字列。
トークンシード	TOTP の計算に使用されるシークレットが表示されます。16 進数または base32 にすることができます。
TOTP 秒	TOTP の生成に使用される時間解決。
TOTP ウィンドウ	トークンの検証時に現在のトークンの前後にチェックされる TOTP トークンの数。
TOTP ピン	特定の時点で TOTP トークンの有効な値。

7.5.2. マルチファクター認証の作成

マルチファクター認証 (MFA) を設定するには、ワンタイムパスワードジェネレーターおよびバックエンド MFA システムが使用するシークレットシードを作成する必要があります。

前提条件

- Linux システム。
- コマンドラインシェルへのアクセス。

手順

1. **urandom** Linux デバイスファイルから 30 文字のシードを生成し、シェル変数 **SEED** に格納します。

   例

   ```
   [user@host01 ~]$ SEED=$(head -10 /dev/urandom | sha512sum | cut -b 1-30)
   ```

2. **SEED** 変数で echo を実行して、シードを出力します。

   例

   ```
   [user@host01 ~]$ echo $SEED
   492dedb20cf51d1405ef6a1316017e
   ```

   同じシードを使用するように、ワンタイムパスワードジェネレーターおよびバックエンドの MFA システムを設定します。

関連情報

- 詳細は、ナレッジベースのソリューション Unable to create RGW MFA token for bucketを参照してください。

- 詳細は、Ceph Object Gateway およびマルチファクター認証 を参照してください。

7.5.3. 新しいマルチファクター認証 TOTP トークンの作成

新たなマルチファクター認証 (MFA) 時間ベースのワンタイムパスワード (TOTP) トークンを作成します。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- Ceph Object Gateway がインストールされている。
- Ceph Monitor ノード上での root アクセスがある。
- ワンタイムパスワードジェネレーターおよび Ceph Object Gateway MFA のシークレットシードが生成されている。

手順

- 新しい MFA TOTP トークンを作成します。

  構文

  ```
  radosgw-admin mfa create --uid=USERID --totp-serial=SERIAL --totp-seed=SEED --totp-seed-type=SEED_TYPE --totp-seconds=TOTP_SECONDS --totp-window=TOTP_WINDOW
  ```

  USERID にはユーザー名を設定し、**SERIAL** には TOTP トークンの ID を表す文字列を設定し、**SEED** には TOTP の計算に使用する 16 進数または base32 値を設定します。以下の設定は任意です。**SEED_TYPE** を **hex** または **base32** に設定し、**TOTP_SECONDS** をタイムアウト (秒単位) に設定するか、**TOTP_WINDOW** を設定して、トークンの検証時に現在のトークンの前後にチェックします。

  例

  ```
  [root@host01 ~]# radosgw-admin mfa create --uid=johndoe --totp-serial=MFAtest --totp-seed=492dedb20cf51d1405ef6a1316017e
  ```

関連情報

- 詳細は、マルチファクター認証用のシードの作成 を参照してください。
- 詳細は、マルチファクター認証トークンの再同期 を参照してください。

7.5.4. マルチファクター認証 TOTP トークンのテスト

マルチファクター認証 (MFA) のタイムベースワンタイムパスワード (TOTP) トークンをテストします。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- Ceph Object Gateway がインストールされている。

- Ceph Monitor ノード上での root アクセスがある。
- MFA TOTP トークンは、**radosgw-admin mfa create** を使用して作成されました。

手順

- TOTP トークンの PIN をテストして、TOTP が正しく機能することを確認します。

  構文

  ```
  radosgw-admin mfa check --uid=USERID --totp-serial=SERIAL --totp-pin=PIN
  ```

  USERID には MFA が設定されているユーザー名を設定し、**SERIAL** には TOTP トークンの ID を表す文字列を設定し、**PIN** にはワンタイムパスワードジェネレーターからの最新の PIN を設定します。

  例

  ```
  [root@host01 ~]# radosgw-admin mfa check  --uid=johndoe --totp-serial=MFAtest --totp-pin=870305
  ok
  ```

  PIN の初回テスト時の場合には、失敗する場合があります。失敗する場合は、トークンを再同期します。**Red Hat Ceph Storage Object Gateway設定および管理ガイド**の **マルチファクター認証トークンの再同期** を参照してください。

関連情報

- 詳細は、**マルチファクター認証用のシードの作成** を参照してください。
- 詳細は、**マルチファクター認証トークンの再同期** を参照してください。

7.5.5. マルチファクター認証 TOTP トークンの再同期

マルチファクター認証 (MFA) のタイムベースのワンタイムパスワードトークンを再同期します。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- Ceph Object Gateway がインストールされている。
- Ceph Monitor ノード上での root アクセスがある。
- MFA TOTP トークンは、**radosgw-admin mfa create** を使用して作成されました。

手順

1. タイムスキューまたはチェックが失敗した場合に、マルチファクター認証 TOTP トークンを再同期します。
   これには、前のピンと現在のピンの 2 回連続して渡す必要があります。

   構文

```
radosgw-admin mfa resync --uid=USERID --totp-serial=SERIAL --totp-pin=PREVIOUS_PIN --totp=pin=CURRENT_PIN
```

USERID には MFA が設定されているユーザー名を設定し、**SERIAL** には TOTP トークンの ID を表す文字列を設定し、**PREVIOUS_PIN** にはユーザーの以前の PIN を設定し、**CURRENT_PIN** にはユーザーの現在の PIN を設定します。

例

```
[root@host01 ~]# radosgw-admin mfa resync --uid=johndoe --totp-serial=MFAtest --totp-pin=802021 --totp-pin=439996
```

2. 新しい PIN をテストしてトークンが正常に再同期されたことを確認します。

構文

```
radosgw-admin mfa check --uid=USERID --totp-serial=SERIAL --totp-pin=PIN
```

USERID には MFA が設定されているユーザー名を設定し、**SERIAL**には TOTP トークンの ID を表す文字列を設定し、**PIN** にはユーザーの PIN を設定します。

例

```
[root@host01 ~]# radosgw-admin mfa check  --uid=johndoe --totp-serial=MFAtest --totp-pin=870305
ok
```

関連情報

- 詳細は、**新しいマルチファクター認証 TOTP トークンの作成** を参照してください。

7.5.6. マルチファクター認証 TOTP トークンのリスト表示

特定のユーザーが持っているすべてのマルチファクター認証 (MFA) 時間ベースのワンタイムパスワード (TOTP) トークンをリスト表示します。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- Ceph Object Gateway がインストールされている。
- Ceph Monitor ノード上での root アクセスがある。
- MFA TOTP トークンは、**radosgw-admin mfa create** を使用して作成されました。

手順

- MFA TOTP トークンをリスト表示します。

構文

```
radosgw-admin mfa list --uid=USERID
```

-

USERID に、MFA が設定されているユーザー名を設定します。

例

```
[root@host01 ~]# radosgw-admin mfa list --uid=johndoe
{
    "entries": [
        {
            "type": 2,
            "id": "MFAtest",
            "seed": "492dedb20cf51d1405ef6a1316017e",
            "seed_type": "hex",
            "time_ofs": 0,
            "step_size": 30,
            "window": 2
        }
    ]
}
```

関連情報

- 詳細は、**新しいマルチファクター認証 TOTP トークンの作成** を参照してください。

7.5.7. マルチファクター認証 TOTP トークンの表示

シリアルを指定して、特定のマルチファクター認証 (MFA) 時間ベースのワンタイムパスワード (TOTP) トークンを表示します。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- Ceph Object Gateway がインストールされている。
- Ceph Monitor ノード上での root アクセスがある。
- MFA TOTP トークンは、**radosgw-admin mfa create** を使用して作成されました。

手順

- MFA TOTP トークンを表示します。

  構文

  ```
  radosgw-admin mfa get --uid=USERID --totp-serial=SERIAL
  ```

  USERID には MFA が設定されているユーザー名に設定し、**SERIAL** には TOTP トークンの ID を表す文字列に設定します。

関連情報

- 詳細は、**新しいマルチファクター認証 TOTP トークンの作成** を参照してください。

7.5.8. マルチファクター認証 TOTP トークンの削除

マルチファクター認証 (MFA) のタイムベースワンタイムパスワード (TOTP) トークンを削除します。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- Ceph Object Gateway がインストールされている。
- Ceph Monitor ノード上での root アクセスがある。
- MFA TOTP トークンは、**radosgw-admin mfa create** を使用して作成されました。

手順

1. MFA TOTP トークンを削除します。

   構文

   ```
   radosgw-admin mfa remove --uid=USERID --totp-serial=SERIAL
   ```

   USERID には MFA が設定されているユーザー名に設定し、**SERIAL** には TOTP トークンの ID を表す文字列に設定します。

   例

   ```
   [root@host01 ~]# radosgw-admin mfa remove --uid=johndoe --totp-serial=MFAtest
   ```

2. MFA TOTP トークンが削除されたことを確認します。

   構文

   ```
   radosgw-admin mfa get --uid=USERID --totp-serial=SERIAL
   ```

   USERID には MFA が設定されているユーザー名に設定し、**SERIAL** には TOTP トークンの ID を表す文字列に設定します。

   例

   ```
   [root@host01 ~]# radosgw-admin mfa get --uid=johndoe --totp-serial=MFAtest
   MFA serial id not found
   ```

関連情報

- 詳細は、Ceph Object Gateway およびマルチファクター認証 を参照してください。

第8章 管理

ストレージ管理者は、**radosgw-admin** コマンドラインインターフェイス (CLI) または Red Hat Ceph Storage Dashboard を使用して Ceph Object Gateway を管理できます。

注記

Red Hat Ceph Storage Dashboard では、すべての Ceph Object Gateway 機能が利用できる訳ではありません。

- ストレージポリシー
- インデックスのないバケット
- バケットインデックスのリシャーディング設定
- 圧縮
- ユーザー管理
- ロールの管理
- クォータの管理
- バケット管理
- 使用方法
- Ceph Object Gateway データレイアウト

前提条件

- 正常かつ実行中の Red Hat Ceph Storage クラスター
- Ceph Object Gateway ソフトウェアのインストール。

8.1. ストレージポリシーの作成

Ceph Object Gateway は配置ターゲットを特定し、配置ターゲットに関連付けられたプールにバケットおよびオブジェクトを保存することで、クライアントバケットとオブジェクトデータを保存します。配置ターゲットを設定しておらず、インスタンスのゾーン設定内のプールにマッピングすると、Ceph Object Gateway はデフォルトのターゲットとプールを使用します (例: **default_placement**)。

ストレージポリシーは、Ceph Object Gateway クライアントに対し、ストレージストラテジーにアクセスする手段を提供します。つまり、たとえば耐久性、レプリケーション、イレイジャーコーディングなどを確保するための方法として、SSD、SAS ドライブ、SATA ドライブなどの特定のタイプのストレージをターゲットに設定する機能があります。詳細は、Red Hat Ceph Storage 7 の **ストレージ戦略** を参照してください。

ストレージポリシーを作成するには、以下の手順に従います。

1. 必要なストレージストラテジーを使用して、新しいプールの **.rgw.buckets.special** を作成します。たとえば、イレイジャーコーディング、特定の CRUSH ルールセット、レプリカ数、および **pg_num** 数および **pgp_num** 数でカスタマイズしたプールなどです。

2. ゾーングループの設定を取得して、これをファイルに保存します。

   構文

   ```
   radosgw-admin zonegroup --rgw-zonegroup=ZONE_GROUP_NAME get > FILE_NAME.json
   ```

   例

   ```
   [root@host01 ~]# radosgw-admin zonegroup --rgw-zonegroup=default get > zonegroup.json
   ```

3. **zonegroup.json** ファイルの **placement_target** の下に、特別な **special-placement** を追加します。

   例

   ```
   {
    "name": "default",
    "api_name": "",
    "is_master": "true",
    "endpoints": [],
    "hostnames": [],
    "master_zone": "",
    "zones": [{
     "name": "default",
     "endpoints": [],
     "log_meta": "false",
     "log_data": "false",
     "bucket_index_max_shards": 5
    }],
    "placement_targets": [{
     "name": "default-placement",
     "tags": []
    }, {
     "name": "special-placement",
     "tags": []
    }],
    "default_placement": "default-placement"
   }
   ```

4. 変更された **zonegroup.json** ファイルでゾーングループを設定します。

   例

   ```
   [root@host01 ~]# radosgw-admin zonegroup set < zonegroup.json
   ```

5. ゾーン設定を取得して、これをファイル (例: **zone.json**) に保存します。

   例

   ```
   [root@host01 ~]# radosgw-admin zone get > zone.json
   ```

6. ゾーンファイルを編集し、**placement_pool** に新しい配置ポリシーキーを追加します。

例

```
{
 "domain_root": ".rgw",
 "control_pool": ".rgw.control",
 "gc_pool": ".rgw.gc",
 "log_pool": ".log",
 "intent_log_pool": ".intent-log",
 "usage_log_pool": ".usage",
 "user_keys_pool": ".users",
 "user_email_pool": ".users.email",
 "user_swift_pool": ".users.swift",
 "user_uid_pool": ".users.uid",
 "system_key": {
  "access_key": "",
  "secret_key": ""
 },
 "placement_pools": [{
  "key": "default-placement",
  "val": {
   "index_pool": ".rgw.buckets.index",
   "data_pool": ".rgw.buckets",
   "data_extra_pool": ".rgw.buckets.extra"
  }
 }, {
  "key": "special-placement",
  "val": {
   "index_pool": ".rgw.buckets.index",
   "data_pool": ".rgw.buckets.special",
   "data_extra_pool": ".rgw.buckets.extra"
  }
 }]
}
```

7. 新しいゾーン設定を設定します。

   例

   ```
   [root@host01 ~]# radosgw-admin zone set < zone.json
   ```

8. ゾーングループのマップを更新します。

   例

   ```
   [root@host01 ~]# radosgw-admin period update --commit
   ```

   special-placement エントリーは **placement_target** としてリスト表示されます。

9. 要求の実行時にストレージポリシーを指定するには、以下を実行します。

   例

   ```
   $ curl -i http://10.0.0.1/swift/v1/TestContainer/file.txt -X PUT -H "X-Storage-Policy: special-placement" -H "X-Auth-Token: AUTH_rgwtxxxxxx"
   ```

8.2. インデックスレスバケットの作成

作成されたバケットがバケットインデックスを使用せずに、オブジェクトのインデックスを格納する、つまりインデックスレスバケットを配置先として設定することができます。データのレプリケーションやリスト表示を使用しない配置ターゲットは、インデックスレスバケットを実装することができます。インデックスレスバケットは、配置ターゲットが特定のバケット内のオブジェクトを追跡しないメカニズムです。これにより、オブジェクト書き込みが発生するたびに発生するリソース競合が削除され、Ceph Object Gateway が Ceph Storage クラスターに必要なラウンドトリップの数を減らします。これにより、同時操作や、小規模のオブジェクト書き込みパフォーマンスに正当な影響を与える可能性があります。

重要

バケットインデックスはバケットの正しい状態を反映せず、これらのバケットをリスト表示してもオブジェクトのリストを正しく返しません。これは複数の機能に影響します。具体的には、バケットインデックスが変更情報の保存に使用されていないため、これらのバケットはマルチゾーン環境では同期されません。この機能にはバケットインデックスが必要になるため、Red Hat は、インデックスレスバケットで S3 オブジェクトバージョン管理を使用することは推奨されません。

注記

インデックスレスバケットを使用すると、単一バケットのオブジェクトの最大数の上限が削除されます。

注記

インデックスレスバケットのオブジェクトは NFS からリスト表示できません。

前提条件

- 実行中で正常な Red Hat Ceph Storage クラスター
- Ceph Object Gateway ソフトウェアのインストール。
- Ceph Object Gateway ノードへのルートレベルのアクセス。

手順

1. 新しい配置ターゲットをゾーングループに追加します。

   例

   ```
   [ceph: root@host03 /]# radosgw-admin zonegroup placement add --rgw-zonegroup="default"
   \
     --placement-id="indexless-placement"
   ```

2. 新しい配置ターゲットをゾーンに追加します。

   例

   ```
   [ceph: root@host03 /]# radosgw-admin zone placement add --rgw-zone="default" \
     --placement-id="indexless-placement" \
     --data-pool="default.rgw.buckets.data" \
   ```

```
    --index-pool="default.rgw.buckets.index" \
    --data_extra_pool="default.rgw.buckets.non-ec" \
    --placement-index-type="indexless"
```

3. ゾーングループのデフォルト配置を **indexless-placement** に設定します。

   例

   ```
   [ceph: root@host03 /]# radosgw-admin zonegroup placement default --placement-id
   "indexless-placement"
   ```

   この例では、**indexless-placement** ターゲットで作成されたバケットはインデックスレスバケットです。

4. クラスターがマルチサイト設定にある場合は、期間を更新し、コミットします。

   例

   ```
   [ceph: root@host03 /]# radosgw-admin period update --commit
   ```

5. 変更を有効にするためには、ストレージクラスター内のすべてのノードで Ceph Object Gateway を再起動します。

   構文

   ```
   ceph orch restart SERVICE_TYPE
   ```

   例

   ```
   [ceph: root@host03 /]# ceph orch restart rgw
   ```

8.3. バケットインデックスのリシャーディングを設定する

ストレージ管理者は、単一サイトおよびマルチサイトデプロイメントでバケットインデックスのリシャーディングを設定して、パフォーマンスを向上させることができます。

手動でオフラインで、または動的にオンラインで、バケットインデックスをリシャーディングできます。

8.3.1. バケットインデックスのリシャーディング

Ceph Object Gateway は、デフォルトで **.rgw.buckets.index** パラメーターに設定されているインデックスプールにバケットインデックスデータを保存します。クライアントが、バケットあたりの最大オブジェクト数のクォータを設定せずに 1 つのバケットに多数のオブジェクトを配置すると、インデックスプールによってパフォーマンスが大幅に低下する可能性があります。

- バケットインデックスのリシャーディングは、バケットごとに多数のオブジェクトを追加する場合のパフォーマンスのボトルネックを防ぎます。
- 新しいバケットのバケットインデックスのリシャーディングを設定したり、既存のバケットのバケットインデックスを変更したりできます。

- 計算されたシャード数に最も近い素数としてシャード数を取得する必要があります。素数であるバケットインデックスシャードは、シャード間で均等に分散されたバケットインデックスエントリーでより適切に機能する傾向があります。

- バケットインデックスは、手動または動的にリシャーディングできます。
バケットインデックスを動的にリシャーディングするプロセス中に、すべての Ceph Object Gateway バケットが定期的にチェックされ、リシャーディングが必要なバケットが検出されます。バケットが **rgw_max_objs_per_shard** パラメーターで指定された値よりも大きい場合、Ceph Object Gateway はバックグラウンドでバケットを動的に再シャードします。**rgw_max_objs_per_shard** のデフォルト値は、シャードごとに 100k オブジェクトです。
バケットインデックスのリシャーディングは、ゾーンまたはゾーングループを変更しなくても、アップグレードされた単一サイト設定で期待どおりに動的に機能します。シングルサイト設定は、以下のいずれかです。

  - レルムのないデフォルトのゾーン設定。
  - レルムが 1 つ以上あるデフォルト以外の設定。
  - 複数レルムのシングルサイト設定。

8.3.2. バケットインデックスの回復

bucket_index_max_shards = 0 で作成されたバケットを再シャーディングすると、バケットのメタデータが削除されます。ただし、影響を受けたバケットを回復することにより、バケットインデックスを復元できます。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- Ceph Object Gateway が少なくとも 2 つのサイトにインストールされている。
- **jq** パッケージがインストールされている。

手順

- バケットインデックスのリカバリーを実行するには、次の 2 つの手順のいずれかを実行します。
  - **radosgw-admin object reindex --bucket BUCKET_NAME --object OBJECT_NAME** コマンドを実行します。
  - スクリプト **/usr/bin/rgw-restore-bucket-index -b BUCKET_NAME -p DATA_POOL_NAME** を実行します。

    例

```
[root@host01 ceph]# /usr/bin/rgw-restore-bucket-index -b bucket-large-1 -p local-zone.rgw.buckets.data

marker is d8a347a4-99b6-4312-a5c1-75b83904b3d4.41610.2
bucket_id is d8a347a4-99b6-4312-a5c1-75b83904b3d4.41610.2
number of bucket index shards is 5
data pool is local-zone.rgw.buckets.data
NOTICE: This tool is currently considered EXPERIMENTAL.
The list of objects that we will attempt to restore can be found in "/tmp/rgwrbi-object-
```

```
list.49946".
Please review the object names in that file (either below or in another window/terminal)
before proceeding.
Type "proceed!" to proceed, "view" to view object list, or "q" to quit: view
Viewing...
Type "proceed!" to proceed, "view" to view object list, or "q" to quit: proceed!
Proceeding...
NOTICE: Bucket stats are currently incorrect. They can be restored with the following
command after 2 minutes:
    radosgw-admin bucket list --bucket=bucket-large-1 --allow-unordered --max-
entries=1073741824
Would you like to take the time to recalculate bucket stats now? [yes/no] yes
Done

real    2m16.530s
user    0m1.082s
sys    0m0.870s
```

注記

- このツールは、バージョン管理されたバケットに対しては機能しません。

```
[root@host01 ~]# time rgw-restore-bucket-index  --proceed serp-bu-ver-1
default.rgw.buckets.data
NOTICE: This tool is currently considered EXPERIMENTAL.
marker is e871fb65-b87f-4c16-a7c3-064b66feb1c4.25076.5
bucket_id is e871fb65-b87f-4c16-a7c3-064b66feb1c4.25076.5
Error: this bucket appears to be versioned, and this tool cannot work with
versioned buckets.
```

- ツールの範囲はシングルサイトのみに限定され、マルチサイトではありません。つまり、サイト 1 で **rgw-restore-bucket-index** ツールを実行しても、サイト 2 のオブジェクトは復元されません。また、その逆も同様です。マルチサイトでは、回復ツールとオブジェクトの再インデックスコマンドはバケットの両方のサイトで実行する必要があります。

8.3.3. バケットインデックスのリシャーディングの制限

重要

注意して、以下の制限を使用してください。お使いのハードウェアの選択には影響があるため、この要件を Red Hat アカウントチームと常に相談してください。

- **リシャーディングが必要になる前の 1 つのバケット内のオブジェクトの最大数:** バケットインデックスシャードごとに最大 102,400 個のオブジェクトを使用します。リシャーディングを最大限に活用して並列処理を最大化するには、Ceph Object Gateway バケットインデックスプールに十分な数の OSD を提供します。この並列化は、Ceph Object Gateway インスタンスの数に応じてスケーリングされ、順序が適切なインデックスシャードの列挙を数列に置き換えます。デフォルトのロックタイムアウトが 60 秒から 90 秒に延長されました。

- **シャード化の使用時の最大オブジェクト数:** 以前のテストに基づき、現在サポートされているバケットインデックスシャードの数は 65521 です。Red Hat の品質保証は、バケットシャーディングで完全なスケーラビリティーテストを実施していません。

- **シャード化の使用時の最大オブジェクト数:** 以前のテストに基づき、現在サポートされているバケットインデックスシャードの数は 65,521 です。
- **他のゾーンが追いつく前にバケットを 3 回再シャーディングできます。**古い世代が同期されるまで、再シャーディングは推奨されません。以前の再シャーディングからの約 4 世代のバケットがサポートされます。制限に達すると、古いログ世代の少なくとも 1 つが完全にトリミングされるまで、動的再シャーディングはバケットを再度再シャーディングしません。コマンド **radosgw-admin bucket reshard** を使用すると、以下のエラーが発生します。

```
Bucket _BUCKET_NAME_ already has too many log generations (4) from previous reshards that peer zones haven't finished syncing.

Resharding is not recommended until the old generations sync, but you can force a reshard with `--yes-i-really-mean-it`.
```

8.3.4. シンプルなデプロイでのバケットインデックスのリシャーディングの設定

すべての新規バケットでバケットインデックスリシャードを有効にし、設定するには、**rgw_override_bucket_index_max_shards** パラメーターを使用します。

パラメーターは以下のいずれかの値に設定できます。

- **0** を指定すると、バケットインデックスのシャーディングが無効になります。これがデフォルト値です。
- **0** より大きい値を有効にすると、バケットシャード化が有効になり、シャードの最大数が設定されます。

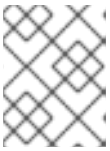
前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- Ceph Object Gateway が少なくとも 2 つのサイトにインストールされている。

手順

1. 推奨されるシャード数を計算します。

```
number of objects expected in a bucket / 100,000
```



注記

現在サポートされているバケットインデックスシャードの最大数は 65,521 です。

2. **rgw_override_bucket_index_max_shards** オプションを適宜設定します。

構文

```
ceph config set client.rgw rgw_override_bucket_index_max_shards VALUE
```

VALUE を、計算されたシャードの推奨数に置き換えます。

例

```
[ceph: root@host01 /]# ceph config set client.rgw rgw_override_bucket_index_max_shards
12
```

- Ceph Object Gateway のすべてのインスタンスに対してバケットインデックスのリシャーディングを設定するには、**rgw_override_bucket_index_max_shards** パラメーターを **global** オプションで設定します。
- Ceph Object Gateway の特定のインスタンスに対してのみバケットインデックスのリシャーディングを設定するには、インスタンスの下に **rgw_override_bucket_index_max_shards** パラメーターを追加します。

3. クラスター内のすべてのノードで Ceph Object Gateways を再起動して、有効にします。

構文

```
ceph orch restart SERVICE_TYPE
```

例

```
[ceph: root#host01 /]# ceph orch restart rgw
```

関連情報

- **バケットインデックスの動的リシャーディング** を参照してください
- **バケットインデックスの手動リシャーディング** を参照してください

8.3.5. マルチサイトデプロイメントでのバケットインデックスのリシャーディングの設定

マルチサイトデプロイメントでは、フェイルオーバーを管理するために、ゾーンごとに異なる **index_pool** 設定を使用できます。1 つのゾーングループ内のゾーンに対して一貫したシャード数を設定するには、そのゾーングループの設定で **bucket_index_max_shards** パラメーターを設定します。**bucket_index_max_shards** パラメーターのデフォルト値は 11 です。

パラメーターは以下のいずれかの値に設定できます。

- バケットインデックスシャード化を無効にする場合は **0**。
- **0** より大きい値を有効にすると、バケットシャード化が有効になり、シャードの最大数が設定されます。

注記

SSD ベースの OSD の CRUSH ルールセットにインデックスプール (該当する場合は各ゾーン) をマッピングすることも、バケットインデックスのパフォーマンスに役立つ可能性があります。詳細は、**パフォーマンスドメインの確立** セクションを参照してください。

重要

マルチサイトデプロイメントで同期の問題を回避するには、バケットに 3 世代を超えるギャップがないようにする必要があります。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- Ceph Object Gateway が少なくとも 2 つのサイトにインストールされている。

手順

1. 推奨されるシャード数を計算します。

```
number of objects expected in a bucket / 100,000
```

注記

現在サポートされているバケットインデックスシャードの最大数は 65,521 です。

2. ゾーングループ設定を **zonegroup.json** ファイルにデプロイメントします。

例

```
[ceph: root@host01 /]# radosgw-admin zonegroup get > zonegroup.json
```

3. **zonegroup.json** ファイルで、名前付きゾーンごとに **bucket_index_max_shards** パラメーターを設定します。

構文

```
bucket_index_max_shards = VALUE
```

VALUE を、計算されたシャードの推奨数に置き換えます。

例

```
bucket_index_max_shards = 12
```

4. ゾーングループをリセットします。

例

```
[ceph: root@host01 /]# radosgw-admin zonegroup set < zonegroup.json
```

5. 期間を更新します。

例

```
[ceph: root@host01 /]# radosgw-admin period update --commit
```

6. リシャーディングが完了したかどうかを確認します。

構文

```
radosgw-admin reshard status --bucket BUCKET_NAME
```

例

```
[ceph: root@host01 /]# radosgw-admin reshard status --bucket data
```

検証

- ストレージクラスターの同期ステータスを確認します。

例

```
[ceph: root@host01 /]# radosgw-admin sync status
```

8.3.6. バケットインデックスの動的リシャーディング

バケットをリシャーディングキューに追加することで、バケットインデックスを動的にリシャーディングできます。リシャーディングされる予定です。リシャードスレッドはバックグラウンドで実行され、スケジュールされたリシャーディングを一度に 1 つずつ実行します。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- Ceph Object Gateway が少なくとも 2 つのサイトにインストールされている。

手順

1. **rgw_dynamic_resharding** パラメーターを **true** に設定します。

例

```
[ceph: root@host01 /]# radosgw-admin period get
```

2. オプション: 次のコマンドを使用して、Ceph 設定をカスタマイズします。

構文

```
ceph config set client.rgw OPTION VALUE
```

OPTION を次のオプションに置き換えます。

- **rgw_reshard_num_logs**: 再シャードログのシャードの数。デフォルト値は **16** です。
- **rgw_reshard_bucket_lock_duration**: リシャード中にバケットのロックの期間。デフォルト値は **360** 秒です。

- **rgw_dynamic_resharding**: 動的リシャードを有効または無効にします。デフォルト値は **true** です。
- **rgw_max_objs_per_shard**: シャードごとのオブジェクトの最大数。デフォルト値は、シャードごとに **100000** オブジェクトです。
- **rgw_reshard_thread_interval**: 再シャード処理のラウンド間の最大時間。デフォルト値は **600** 秒です。

例

```
[ceph: root@host01 /]# ceph config set client.rgw rgw_reshard_num_logs 23
```

3. リシャーディングキューにバケットを追加します。

構文

```
radosgw-admin reshard add --bucket BUCKET --num-shards NUMBER
```

例

```
[ceph: root@host01 /]# radosgw-admin reshard add --bucket data --num-shards 10
```

4. リシャーディングキューを一覧表示します。

例

```
[ceph: root@host01 /]# radosgw-admin reshard list
```

5. バケットログの世代とシャードを確認します。

例

```
[ceph: root@host01 /]# radosgw-admin bucket layout --bucket data
{
    "layout": {
        "resharding": "None",
        "current_index": {
            "gen": 1,
            "layout": {
                "type": "Normal",
                "normal": {
                    "num_shards": 23,
                    "hash_type": "Mod"
                }
            }
        },
        "logs": [
            {
                "gen": 0,
                "layout": {
                    "type": "InIndex",
                    "in_index": {
                        "gen": 0,
```

```
                    "layout": {
                        "num_shards": 11,
                        "hash_type": "Mod"
                    }
                }
            }
        },
        {
            "gen": 1,
            "layout": {
                "type": "InIndex",
                "in_index": {
                    "gen": 1,
                    "layout": {
                        "num_shards": 23,
                        "hash_type": "Mod"
                    }
                }
            }
        }
    ]
  }
}
```

6. バケットの再シャーディングステータスを確認するには、以下のコマンドを実行します。

   構文

   ```
   radosgw-admin reshard status --bucket BUCKET
   ```

   例

   ```
   [ceph: root@host01 /]# radosgw-admin reshard status --bucket data
   ```

7. リシャーディングキューのエントリーをすぐに処理します。

   ```
   [ceph: root@host01 /]# radosgw-admin reshard process
   ```

8. 保留中のバケットのリシャーディングをキャンセルします:

   警告

   保留中 の再シャード操作のみをキャンセルできます。**継続中** の再シャード操作をキャンセルしないでください。

   構文

   ```
   radosgw-admin reshard cancel --bucket BUCKET
   ```

例

```
[ceph: root@host01 /]# radosgw-admin reshard cancel --bucket data
```

検証

- バケットの再シャーディングステータスを確認するには、以下のコマンドを実行します。

  構文

  ```
  radosgw-admin reshard status --bucket BUCKET
  ```

  例

  ```
  [ceph: root@host01 /]# radosgw-admin reshard status --bucket data
  ```

関連情報

- 古いバケットエントリーを削除するには、**リシャーディング後のバケットエントリーの古いインスタンスのクリーニング** セクションを参照してください。
- **バケットインデックスの手動リシャーディング** を参照してください。
- **シンプルなデプロイでのバケットインデックスのリシャーディングの設定** を参照してください。

8.3.7. マルチサイト設定でバケットインデックスを動的にリシャーディングする

Red Hat Ceph Storage は、マルチサイト設定でのバケットインデックスの動的再シャーディングをサポートします。この機能により、オブジェクトのレプリケーションを中断せずに、マルチサイト設定でバケットを再シャーディングできます。**rgw_dynamic_resharding** を有効にすると、各ゾーンで個別に実行され、ゾーンで同じバケットに異なるシャード数が選択される可能性があります。

従う必要があるこれらの手順は、既存の Red Hat Ceph Storage クラスター **専用** です。ストレージクラスターのアップグレード後に、既存のゾーンおよびゾーングループで **resharding** 機能を手動で有効にする必要があります。

注記

Red Hat Ceph Storage 6.0 の新規インストールでは、ゾーンおよびゾーングループの **resharding** 機能がデフォルトでサポートされ、有効にされます。

注記

他のゾーンが追い付く前に、バケットを 3 回再シャーディングできます。詳細は、**バケットインデックスのリシャーディングの制限** を参照してください。

注記

バケットが作成され、動的にリシャーディングするためのオブジェクト数のしきい値を超えてアップロードされた場合、リシャーディングプロセスを開始するには、引き続き古いバケットに I/O を書き込む必要があります。

前提条件

- 両方のサイトの Red Hat Ceph Storage クラスターは、最新バージョンにアップグレードされている。
- 両方のサイトで有効になっているすべての Ceph Object Gateway デーモンは、最新バージョンにアップグレードされている。
- すべてのノードへの root レベルのアクセス。

手順

1. ゾーングループで **resharding** が有効になっているかどうかを確認します。

   例

   ```
   [ceph: root@host01 /]# radosgw-admin sync status
   ```

   ゾーングループでの再シャーディング用に **zonegroup features enabled** が有効化されていない場合は、手順を続行してください。

2. Ceph Object Gateway がインストールされているマルチサイト設定のすべてのゾーングループで、**resharding** 機能を有効にします。

   構文

   ```
   radosgw-admin zonegroup modify --rgw-zonegroup=ZONEGROUP_NAME --enable-feature=resharding
   ```

   例

   ```
   [ceph: root@host01 /]# radosgw-admin zonegroup modify --rgw-zonegroup=us --enable-feature=resharding
   ```

3. 期間を更新してコミットします。

   例

   ```
   [ceph: root@host01 /]# radosgw-admin period update --commit
   ```

4. Ceph Object Gateway がインストールされているマルチサイト設定のすべてのゾーンで、**resharding** 機能を有効にします。

   構文

   ```
   radosgw-admin zone modify --rgw-zone=ZONE_NAME --enable-feature=resharding
   ```

   例

   ```
   [ceph: root@host01 /]# radosgw-admin zone modify --rgw-zone=us-east --enable-feature=resharding
   ```

5. 期間を更新してコミットします。

例

```
[ceph: root@host01 /]# radosgw-admin period update --commit
```

6. ゾーンおよびゾーングループで、**resharding** 機能が有効化されていることを確認します。各ゾーンが **supported_features** をリスト表示し、ゾーングループで **enabled_features** がリストされていることを確認できます。

例

```
[ceph: root@host01 /]# radosgw-admin period get

"zones": [
                {
                    "id": "505b48db-6de0-45d5-8208-8c98f7b1278d",
                    "name": "us_east",
                    "endpoints": [
                        "http://10.0.208.11:8080"
                    ],
                    "log_meta": "false",
                    "log_data": "true",
                    "bucket_index_max_shards": 11,
                    "read_only": "false",
                    "tier_type": "",
                    "sync_from_all": "true",
                    "sync_from": [],
                    "redirect_zone": "",
                    "supported_features": [
                        "resharding"
                    ]
            "default_placement": "default-placement",
            "realm_id": "26cf6f23-c3a0-4d57-aae4-9b0010ee55cc",
            "sync_policy": {
                "groups": []
            },
            "enabled_features": [
                "resharding"
            ]
```

7. 同期のステータスを確認します。

例

```
[ceph: root@host01 /]# radosgw-admin sync status
          realm 26cf6f23-c3a0-4d57-aae4-9b0010ee55cc (usa)
      zonegroup 33a17718-6c77-493e-99fe-048d3110a06e (us)
           zone 505b48db-6de0-45d5-8208-8c98f7b1278d (us_east)
zonegroup features enabled: resharding
```

この例では、**resharding** 機能が **us** ゾーングループに対して有効になっています。

8. オプション: ゾーングループの **resharding** 機能を無効にすることができます。

重要

任意の単一ゾーンで再シャーディングを無効にするには、その特定のゾーンで **rgw_dynamic_resharding** 設定オプションを **false** に設定します。

a. Ceph Object Gateway がインストールされているマルチサイトのすべてのゾーングループで、機能を無効にします。

構文

```
radosgw-admin zonegroup modify --rgw-zonegroup=ZONEGROUP_NAME --disable-feature=resharding
```

例

```
[ceph: root@host01 /]# radosgw-admin zonegroup modify --rgw-zonegroup=us --disable-feature=resharding
```

b. 期間を更新してコミットします。

例

```
[ceph: root@host01 /]# radosgw-admin period update --commit
```

関連情報

- 動的バケットインデックスの再シャーディングの設定可能なパラメーターの詳細は、**Red Hat Ceph Storage オブジェクトゲートウェイ設定および管理ガイド** の **バケットインデックスの動的再シャーディング** セクションを参照してください。

8.3.8. バケットインデックスの手動リシャーディング

バケットが最適化された初期設定よりも大きくなった場合は、**radosgw-admin bucket reshard** コマンドを使用してバケットインデックスプールをリシャーディングします。このコマンドは、次のタスクを実行します。

- 指定されたバケットのバケットインデックスオブジェクトの新しいセットを作成します。
- これらのバケットインデックスオブジェクトにオブジェクトエントリーを分散します。
- 新規バケットインスタンスを作成します。
- 新規インデックス操作すべてが新規バケットインデックスを通過するように、新しいバケットインスタンスをバケットとリンクします。
- 古いバケット ID および新しいバケット ID をコマンド出力に出力します。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- Ceph Object Gateway が少なくとも 2 つのサイトにインストールされている。

手順

1. 元のバケットインデックスをバックアップします。

   構文

   ```
   radosgw-admin bi list --bucket=BUCKET > BUCKET.list.backup
   ```

   例

   ```
   [ceph: root@host01 /]# radosgw-admin bi list --bucket=data > data.list.backup
   ```

2. バケットインデックスを再シャード化します。

   構文

   ```
   radosgw-admin bucket reshard --bucket=BUCKET --num-shards=NUMBER
   ```

   例

   ```
   [ceph: root@host01 /]# radosgw-admin bucket reshard --bucket=data --num-shards=100
   ```

検証

- バケットの再シャーディングステータスを確認するには、以下のコマンドを実行します。

  構文

  ```
  radosgw-admin reshard status --bucket bucket
  ```

  例

  ```
  [ceph: root@host01 /]# radosgw-admin reshard status --bucket data
  ```

関連情報

- 詳細は、**Red Hat Ceph Storage Object Gateway ガイド** の **マルチサイトデプロイメントでのバケットインデックスのリシャーディングの設定** を参照してください。
- **バケットインデックスの動的リシャーディング** を参照してください
- **シンプルなデプロイでのバケットインデックスのリシャーディングの設定** を参照してください。

8.3.9. リシャーディング後のバケットエントリーの古いインスタンスのクリーニング

リシャーディングプロセスでは、バケットエントリーの古いインスタンスが自動的に消去されない場合があり、これらのインスタンスはストレージクラスターのパフォーマンスに影響を与える可能性があります。

古いインスタンスが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ないように、手動でクリーンアップします。

重要

古いインスタンスを消去する前に、Red Hat サポート にお問い合わせください。

重要

この手順は、マルチサイトクラスターではなく、単純なデプロイメントでのみ使用してください。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- Ceph Object Gateway がインストールされている。

手順

1. 古いインスタンスをリスト表示します。

```
[ceph: root@host01 /]# radosgw-admin reshard stale-instances list
```

2. バケットエントリーの古いインスタンスを消去します。

```
[ceph: root@host01 /]# radosgw-admin reshard stale-instances rm
```

検証

- バケットの再シャーディングステータスを確認するには、以下のコマンドを実行します。

  構文

```
radosgw-admin reshard status --bucket BUCKET
```

  例

```
[ceph: root@host01 /]# radosgw-admin reshard status --bucket data
```

8.3.10. 圧縮の有効化

Ceph Object Gateway は、Ceph の圧縮プラグインを使用してアップロードしたオブジェクトのサーバー側の圧縮をサポートします。これには、以下が含まれます。

- **zlib**: サポート対象。
- **snappy**: サポート対象。
- **zstd**: サポート対象。

設定

ゾーンの配置ターゲットで圧縮を有効にするには、**--compression=TYPE** オプションを **radosgw-admin zone placement modify** コマンドに指定します。圧縮 **TYPE** は、新しいオブジェクトデータの書き込み時に使用する圧縮プラグインの名前を指します。

圧縮される各オブジェクトは圧縮タイプを保存します。設定を変更しても、既存の圧縮オブジェクトをデプロイメントします。また、Ceph Object Gateway は強制的に既存オブジェクトを再圧縮する訳ではありません。

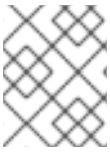
この圧縮設定は、この配置ターゲットを使用してバケットにアップロードされるすべての新規オブジェクトに適用されます。

ゾーンの配置ターゲットで圧縮を無効にするには、 **--compression=TYPE** オプションを **radosgw-admin zone placement modify** コマンドに指定して、空の文字列または **none** を指定します。

例

```
[root@host01 ~] radosgw-admin zone placement modify --rgw-zone=default --placement-id=default-placement --compression=zlib
{
...
    "placement_pools": [
        {
            "key": "default-placement",
            "val": {
                "index_pool": "default.rgw.buckets.index",
                "data_pool": "default.rgw.buckets.data",
                "data_extra_pool": "default.rgw.buckets.non-ec",
                "index_type": 0,
                "compression": "zlib"
            }
        }
    ],
...
}
```

圧縮の有効化または無効化後に Ceph Object Gateway インスタンスを再起動して、変更を反映します。



注記

Ceph Object Gateway は **デフォルト** ゾーンとプールのセットを作成します。実稼働デプロイメントの場合は、**レルムの作成** セクションを最初に参照してください。

統計

既存のコマンドおよび API は、圧縮されていないデータに基づいてオブジェクトおよびバケットサイズを引き続き報告しますが、**radosgw-admin bucket stats** コマンドにはすべてのバケットの圧縮統計が含まれます。

radosgw-adminbucket stats コマンドの使用タイプは次のとおりです。

- **rgw.main** は、通常のエントリーまたはオブジェクトを指します。
- **rgw.multimeta** は、不完全なマルチパートアップロードのメタデータを指します。
- **rgw.cloudtiered** は、ライフサイクルポリシーによってクラウド層に移行されたオブジェクトを指します。**restart_head_object=true** で設定すると、データを含まないヘッドオブジェクトが残されますが、それでも HeadObject リクエストを介してオブジェクトのメタデータを提供

できます。これらのスタブヘッドオブジェクトは、**rgw.cloudtiered** カテゴリーを使用します。詳細は、**Red Hat Ceph Storage オブジェクトゲートウェイガイド** の Amazon S3 クラウドサービスへのデータの移行 セクションを参照してください。

構文

```
radosgw-admin bucket stats --bucket=BUCKET_NAME
{
...
    "usage": {
        "rgw.main": {
            "size": 1075028,
            "size_actual": 1331200,
            "size_utilized": 592035,
            "size_kb": 1050,
            "size_kb_actual": 1300,
            "size_kb_utilized": 579,
            "num_objects": 104
        }
    },
...
}
```

size は、バケット内の非圧縮および非暗号化のオブジェクトの累積サイズです。**size_kb** はキロバイト単位の累積サイズであり、**ceiling(size/1024)** として計算されます。この例では、**ceiling(1075028/1024) = 1050** です。

size_actual は、各オブジェクトが 4096 バイトのブロックのセットに分散されてからの全オブジェクトの累積サイズです。バケットに 2 つのオブジェクト (1 つはサイズ 4100 バイト、もう 1 つは 8500 バイト) がある場合、最初のオブジェクトは 8192 バイトに切り上げられ、2 番目のオブジェクトは 12288 バイトに切り上げられ、バケットの合計は 20480 バイトになります。**size_kb_actual** はキロバイト単位の実際のサイズで、**size_actual/1024** として計算されます。上記の例では、**1331200/1024 = 1300** になります。

size_utilized は、圧縮または暗号化、もしくはその両方が行われた後のデータの合計サイズ (バイト単位) です。暗号化するとオブジェクトのサイズが増加する可能性がありますが、圧縮するとオブジェクトのサイズが減少する可能性があります。**size_kb_utilized** はキロバイト単位の合計サイズで、**ceiling(size_utilized/1024)** として計算されます。この例では、**ceiling(592035/1024)= 579** です。

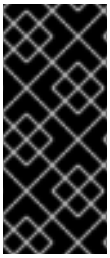
8.4. ユーザー管理

Ceph Object Storage ユーザー管理とは、Ceph Storage Cluster のクライアントアプリケーションとしての Ceph Object Gateway ではなく、Ceph Object Storage サービスのクライアントアプリケーションであるユーザーを指します。クライアントアプリケーションが Ceph Object Gateway サービスと対話できるようにするには、ユーザー、アクセスキー、およびシークレットを作成する必要があります。

ユーザータイプが 2 つあります。

- **User:** user という用語は、S3 インターフェイスのユーザーを反映しています。
- **Subuser:** subuser という用語は、Swift インターフェイスのユーザーを反映しています。サブユーザーがユーザーに関連付けられています。

ユーザーとサブユーザーを作成、変更、表示、一時停止、および削除できます。



重要

マルチサイトデプロイメントでユーザーを管理する場合は、常にマスターゾーングループのマスターゾーン内の Ceph Object Gateway ノードで **radosgw-admin** コマンドを発行して、ユーザーがマルチサイトクラスター全体で同期するようにします。マルチサイトクラスター上のユーザーをセカンダリーゾーンまたはセカンダリーゾーングループから作成、変更、または削除しないでください。

ユーザーおよびサブユーザー ID の作成に加え、ユーザーの表示名およびメールアドレスを追加することができます。キーおよびシークレットを指定するか、キーおよびシークレットを自動的に生成できます。キーを生成または指定した際には、ユーザー ID が S3 キータイプに対応し、サブユーザー ID が Swift キータイプに対応することに注意してください。Swift キーには、アクセスレベルの **read**、**write**、**readwrite**、および **full** もあります。

ユーザー管理コマンドライン構文は、通常、パターン **user COMMAND USER_ID** に従います。ここで、**USER_ID** は、**--uid=** オプションの後にユーザー ID (S3) が続くか、 **--subuser=** オプションの後にユーザー名 (Swift) が続きます。

構文

```
radosgw-admin user <create|modify|info|rm|suspend|enable|check|stats> <--uid=USER_ID|--subuser=SUB_USER_NAME> [other-options]
```

発行するコマンドによっては、追加のオプションが必要になる場合があります。

8.4.1. マルチテナンシー

Ceph Object Gateway は S3 および Swift API の両方に対するマルチテナンシーをサポートします。この場合、各ユーザーとバケットはテナント下に置かれます。 マルチテナンシーは、複数のテナントが共通のバケット名 (例: test、main など) を使用している場合に、名前空間のクラッシュを防ぎます。

各ユーザーとバケットはテナントの下にあります。下位互換性のために、空の名前を持つレガシーテナントが追加されます。テナントを具体的に指定せずにバケットを参照する場合は常に、Swift API はレガシーテナントを想定します。既存のユーザーもレガシーテナントに保存されるため、以前のリリースと同様にバケットとオブジェクトにアクセスします。

このようなテナントの場合、テナント自体には何の操作もありません。ユーザーが管理されている場合には、必要に応じて表示および非表示になります。明示的なテナントを持つユーザーを作成、変更、および削除するには、追加のオプション **--tenant** を指定するか、**radosgw-admin** コマンドのパラメーターで構文 **"TENANT$USER"** を使用します。

S3 用のユーザー **testx$tester** を作成するには、以下のコマンドを実行します。

例

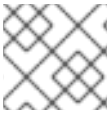
```
[root@host01 ~]# radosgw-admin --tenant testx --uid tester \
                --display-name "Test User" --access_key TESTER \
                --secret test123 user create
```

Swift のユーザー **testx$tester** を作成するには、以下のコマンドのいずれかを実行します。

例

```
[root@host01 ~]# radosgw-admin --tenant testx --uid tester \
```

```
                --display-name "Test User" --subuser tester:swift \
                --key-type swift --access full subuser create

[root@host01 ~]# radosgw-admin key create --subuser 'testx$tester:swift' \
                --key-type swift --secret test123
```



注記

明示的なテナントを持つサブユーザーは、シェルで引用する必要がありました。

8.4.2. ユーザーを作成します。

user create コマンドを使用して S3-interface ユーザーを作成します。ユーザー ID と表示名を指定する必要があります。メールアドレスを指定することもできます。key または secret を指定しないと、**radosgw-admin** によって自動的に生成されます。ただし、生成されたキー/シークレットのペアを使用しない場合は、キーやシークレットを指定できます。

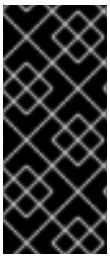
構文

```
radosgw-admin user create --uid=USER_ID \
[--key-type=KEY_TYPE] [--gen-access-key|--access-key=ACCESS_KEY]\
[--gen-secret | --secret=SECRET_KEY] \
[--email=EMAIL] --display-name=DISPLAY_NAME
```

例

```
[root@host01 ~]# radosgw-admin user create --uid=janedoe --access-
key=11BS02LGFB6AL6H1ADMW --secret=vzCEkuryfn060dfee4fgQPqFrncKEIkh3ZcdOANY --
email=jane@example.com --display-name=Jane Doe
```

```
{ "user_id": "janedoe",
  "display_name": "Jane Doe",
  "email": "jane@example.com",
  "suspended": 0,
  "max_buckets": 1000,
  "auid": 0,
  "subusers": [],
  "keys": [
        { "user": "janedoe",
          "access_key": "11BS02LGFB6AL6H1ADMW",
          "secret_key": "vzCEkuryfn060dfee4fgQPqFrncKEIkh3ZcdOANY"}],
  "swift_keys": [],
  "caps": [],
  "op_mask": "read, write, delete",
  "default_placement": "",
  "placement_tags": [],
  "bucket_quota": { "enabled": false,
      "max_size_kb": -1,
      "max_objects": -1},
  "user_quota": { "enabled": false,
      "max_size_kb": -1,
      "max_objects": -1},
  "temp_url_keys": []}
```



重要

キーの出力を確認します。**radosgw-admin** が JSON エスケープ (\) 文字を生成することがあり、一部のクライアントは JSON エスケープ文字の処理方法を知りません。対処法には、JSON エスケープ文字 (\) の削除、文字列の引用符でのカプセル化、キーの再生成、JSON エスケープ文字が含まれていないことの確認、またはキーとシークレットの手動指定が含まれます。

8.4.3. サブユーザーの作成

サブユーザー (Swift インターフェイス) を作成するには、ユーザー ID (**--uid=USERNAME**)、サブユーザー ID、およびサブユーザーのアクセスレベルを指定する必要があります。key または secret を指定しないと、**radosgw-admin** によって自動的に生成されます。ただし、生成されたキー/シークレットのペアを使用しない場合は、キーかシークレットまたはその両方を指定できます。

注記

アクセス制御ポリシーも含まれるため、**full** は **readwrite** ではありません。

構文

```
radosgw-admin subuser create --uid=USER_ID --subuser=SUB_USER_ID --access=[ read | write | readwrite | full ]
```

例

```
[root@host01 ~]# radosgw-admin subuser create --uid=janedoe --subuser=janedoe:swift --access=full

{ "user_id": "janedoe",
  "display_name": "Jane Doe",
  "email": "jane@example.com",
  "suspended": 0,
  "max_buckets": 1000,
  "auid": 0,
  "subusers": [
        { "id": "janedoe:swift",
          "permissions": "full-control"}],
  "keys": [
        { "user": "janedoe",
          "access_key": "11BS02LGFB6AL6H1ADMW",
          "secret_key": "vzCEkuryfn060dfee4fgQPqFrncKEIkh3ZcdOANY"}],
  "swift_keys": [],
  "caps": [],
  "op_mask": "read, write, delete",
  "default_placement": "",
  "placement_tags": [],
  "bucket_quota": { "enabled": false,
      "max_size_kb": -1,
      "max_objects": -1},
  "user_quota": { "enabled": false,
      "max_size_kb": -1,
      "max_objects": -1},
  "temp_url_keys": []}
```

8.4.4. ユーザー情報の取得

ユーザーに関する情報を取得するには、**user info** ユーザー ID (**--uid=USERNAME**) を指定します。

例

```
[root@host01 ~]# radosgw-admin user info --uid=janedoe
```

テナントユーザーに関する情報を取得するには、ユーザー ID とテナントの名前の両方を指定します。

```
[root@host01 ~]# radosgw-admin user info --uid=janedoe --tenant=test
```

8.4.5. ユーザー情報の変更

ユーザーに関する情報を変更するには、ユーザー ID (**--uid=USERNAME**) と変更する属性を指定する必要があります。変更は通常、キーとシークレット、電子メールアドレス、表示名、およびアクセスレベルに対して行われます。

例

```
[root@host01 ~]# radosgw-admin user modify --uid=janedoe --display-name="Jane E. Doe"
```

サブユーザーの値を変更するには、**subuser modify** とサブユーザー ID を指定します。

例

```
[root@host01 ~]# radosgw-admin subuser modify --subuser=janedoe:swift --access=full
```

8.4.6. ユーザーの有効化および一時停止

ユーザーを作成すると、ユーザーはデフォルトで有効になります。ただし、ユーザー特権を一時停止して、後で再度有効にすることができます。ユーザーを一時停止するには、**user suspend** とユーザー ID を指定します。

```
[root@host01 ~]# radosgw-admin user suspend --uid=johndoe
```

一時停止ユーザーを再度有効にするには、**user enable** とユーザー ID を指定します。

```
[root@host01 ~]# radosgw-admin user enable --uid=johndoe
```

注記

ユーザーを無効にすると、サブユーザーが無効になります。

8.4.7. ユーザーの削除

ユーザーを削除すると、ユーザーとサブユーザーはシステムから削除されます。ただし、必要に応じてサブユーザーのみを削除できます。ユーザー (およびサブユーザー) を削除するには、**user rm** とユーザー ID を指定します。

構文

```
radosgw-admin user rm --uid=USER_ID[--purge-keys] [--purge-data]
```

例

```
[ceph: root@host01 /]# radosgw-admin user rm --uid=johndoe --purge-data
```

サブユーザーのみを削除するには、**subuser rm** およびサブユーザー名を指定します。

例

```
[ceph: root@host01 /]# radosgw-admin subuser rm --subuser=johndoe:swift --purge-keys
```

オプションには以下が含まれます。

- **データのパージ: --purge-data** オプションは、UID に関連付けられたすべてのデータをパージします。
- **キーのパージ: --purge-keys** オプションは、UID に関連付けられたすべてのキーをパージします。

8.4.8. サブユーザーの削除

サブユーザーを削除すると、Swift インターフェイスへのアクセスが削除されます。ユーザーがシステムに残ります。サブユーザーを削除するには、**subuser rm** およびサブユーザー ID を指定します。

構文

```
radosgw-admin subuser rm --subuser=SUB_USER_ID
```

例

```
[root@host01 /]# radosgw-admin subuser rm --subuser=johndoe:swift
```

オプションには以下が含まれます。

- **キーのパージ: --purge-keys** オプションは、UID に関連付けられたすべてのキーをパージします。

8.4.9. ユーザーの名前を変更します。

ユーザーの名前を変更するには、**radosgw-admin user rename** コマンドを使用します。このコマンドにかかる時間は、ユーザーが持つバケットおよびオブジェクトの数によって異なります。この数字が大きい場合、Red Hat は、**screen** パッケージが提供する **Screen** ユーティリティーでコマンドを使用することを推奨します。

前提条件

- 稼働中の Ceph クラスター。
- Ceph Object Gateway を実行しているホストへの **root** または **sudo** アクセス。
- インストールされた Ceph Object Gateway。

手順

1. ユーザーの名前を変更します。

構文

```
radosgw-admin user rename --uid=CURRENT_USER_NAME --new-
uid=NEW_USER_NAME
```

例

```
[ceph: root@host01 /]# radosgw-admin user rename --uid=user1 --new-uid=user2

{
    "user_id": "user2",
    "display_name": "user 2",
    "email": "",
    "suspended": 0,
    "max_buckets": 1000,
    "auid": 0,
    "subusers": [],
    "keys": [
        {
            "user": "user2",
            "access_key": "59EKHI6AI9F8WOW8JQZJ",
            "secret_key": "XH0uY3rKCUcuL73X0ftjXbZqUbk0cavD11rD8MsA"
        }
    ],
    "swift_keys": [],
    "caps": [],
    "op_mask": "read, write, delete",
    "default_placement": "",
    "placement_tags": [],
    "bucket_quota": {
        "enabled": false,
        "check_on_raw": false,
        "max_size": -1,
        "max_size_kb": 0,
        "max_objects": -1
    },
    "user_quota": {
        "enabled": false,
        "check_on_raw": false,
        "max_size": -1,
        "max_size_kb": 0,
        "max_objects": -1
    },
    "temp_url_keys": [],
    "type": "rgw"
}
```

ユーザーがテナント内にある場合は、ユーザー名とテナントの両方を指定します。

構文

```
radosgw-admin user rename --uid USER_NAME --new-uid NEW_USER_NAME --tenant
TENANT
```

例

```
[ceph: root@host01 /]# radosgw-admin user rename --uid=test$user1 --new-uid=test$user2 -
-tenant test

1000 objects processed in tvtester1. Next marker 80_tVtester1_99
2000 objects processed in tvtester1. Next marker 64_tVtester1_44
3000 objects processed in tvtester1. Next marker 48_tVtester1_28
4000 objects processed in tvtester1. Next marker 2_tVtester1_74
5000 objects processed in tvtester1. Next marker 14_tVtester1_53
6000 objects processed in tvtester1. Next marker 87_tVtester1_61
7000 objects processed in tvtester1. Next marker 6_tVtester1_57
8000 objects processed in tvtester1. Next marker 52_tVtester1_91
9000 objects processed in tvtester1. Next marker 34_tVtester1_74
9900 objects processed in tvtester1. Next marker 9_tVtester1_95
1000 objects processed in tvtester2. Next marker 82_tVtester2_93
2000 objects processed in tvtester2. Next marker 64_tVtester2_9
3000 objects processed in tvtester2. Next marker 48_tVtester2_22
4000 objects processed in tvtester2. Next marker 32_tVtester2_42
5000 objects processed in tvtester2. Next marker 16_tVtester2_36
6000 objects processed in tvtester2. Next marker 89_tVtester2_46
7000 objects processed in tvtester2. Next marker 70_tVtester2_78
8000 objects processed in tvtester2. Next marker 51_tVtester2_41
9000 objects processed in tvtester2. Next marker 33_tVtester2_32
9900 objects processed in tvtester2. Next marker 9_tVtester2_83
{
    "user_id": "test$user2",
    "display_name": "User 2",
    "email": "",
    "suspended": 0,
    "max_buckets": 1000,
    "auid": 0,
    "subusers": [],
    "keys": [
        {
            "user": "test$user2",
            "access_key": "user2",
            "secret_key": "123456789"
        }
    ],
    "swift_keys": [],
    "caps": [],
    "op_mask": "read, write, delete",
    "default_placement": "",
    "placement_tags": [],
    "bucket_quota": {
        "enabled": false,
        "check_on_raw": false,
        "max_size": -1,
        "max_size_kb": 0,
        "max_objects": -1
    },
```

```
    "user_quota": {
        "enabled": false,
        "check_on_raw": false,
        "max_size": -1,
        "max_size_kb": 0,
        "max_objects": -1
    },
    "temp_url_keys": [],
    "type": "rgw"
}
```

2. ユーザーの名前が正常に変更されたことを確認します。

構文

```
radosgw-admin user info --uid=NEW_USER_NAME
```

例

```
[ceph: root@host01 /]# radosgw-admin user info --uid=user2
```

ユーザーがテナント内にある場合は、**TENANT$USER_NAME** 形式を使用します。

構文

```
radosgw-admin user info --uid= TENANT$USER_NAME
```

例

```
[ceph: root@host01 /]# radosgw-admin user info --uid=test$user2
```

関連情報

- man ページの **screen(1)**

8.4.10. キーの作成

ユーザーのキーを作成するには、**key create** を指定する必要があります。ユーザーには、ユーザー ID と **s3** キータイプを指定します。サブユーザーのキーを作成するには、サブユーザー ID と **swift** キータイプを指定する必要があります。

例

```
[ceph: root@host01 /]# radosgw-admin key create --subuser=johndoe:swift --key-type=swift --gen-secret

{ "user_id": "johndoe",
  "rados_uid": 0,
  "display_name": "John Doe",
  "email": "john@example.com",
  "suspended": 0,
  "subusers": [
    { "id": "johndoe:swift",
```

```
      "permissions": "full-control"}],
  "keys": [
    { "user": "johndoe",
      "access_key": "QFAMEDSJP5DEKJO0DDXY",
      "secret_key": "iaSFLDVvDdQt6lkNzHyW4fPLZugBAI1g17LO0+87"}],
  "swift_keys": [
    { "user": "johndoe:swift",
      "secret_key": "E9T2rUZNu2gxUjcwUBO8n\/Ev4KX6\/GprEuH4qhu1"}]}
```

8.4.11. アクセスキーの追加および削除

ユーザーおよびサブユーザーには、S3 インターフェイスおよび Swift インターフェイスを使用するためのアクセスキーが必要です。ユーザーまたはサブユーザーを作成し、アクセスキーおよびシークレットを指定しない場合、キーおよびシークレットが自動的に生成されます。キーを作成し、アクセスキーやシークレットを指定または生成することができます。アクセスキーおよびシークレットを削除することもできます。オプションには以下が含まれます。

- **--secret=SECRET_KEY** は、たとえば手動で生成されたシークレットキーを指定します。
- **--geen-access-key** は、ランダムなアクセスキーを生成します (デフォルトでは S3 ユーザー用)。
- **--geen-secret** は、ランダムな秘密鍵を生成します。
- **--key-type=KEY_TYPE** は、キータイプを指定します。オプションは swift および s3 です。

キーを追加するには、ユーザーを指定します。

例

```
[root@host01 ~]# radosgw-admin key create --uid=johndoe --key-type=s3 --gen-access-key --gen-secret
```

鍵とシークレットを指定することもできます。

アクセスキーを削除するには、ユーザーとキーを指定する必要があります。

1. 特定ユーザーのアクセスキーを検索します。

   例

   ```
   [root@host01 ~]# radosgw-admin user info --uid=johndoe
   ```

   アクセスキーは、出力の **"access_key"** 値になります。

   例

   ```
   [root@host01 ~]# radosgw-admin user info --uid=johndoe
   {
       "user_id": "johndoe",
       ...
       "keys": [
           {
               "user": "johndoe",
               "access_key": "0555b35654ad1656d804",
   ```

```
            "secret_key":
"h7GhxuBLTrlhVUyxSPUKUV8r/2EI4ngqJxD7iBdBYLhwluN30JaT3Q=="
        }
    ],
    ...
}
```

2. 前の手順のユーザー ID とアクセスキーを指定して、アクセスキーを削除します。

構文

```
radosgw-admin key rm --uid=USER_ID --access-key ACCESS_KEY
```

例

```
[root@host01 ~]# radosgw-admin key rm --uid=johndoe --access-key
0555b35654ad1656d804
```

8.4.12. 管理機能の追加および削除

Ceph Storage Cluster は、ユーザーが REST API を介して管理機能を実行できるようにする管理 API を提供します。デフォルトでは、ユーザーはこの API にアクセスできません。ユーザーが管理機能を実行できるようにするには、ユーザーに管理機能を提供します。

ユーザーに管理ケイパビリティーを追加するには、以下のコマンドを実行します。

構文

```
radosgw-admin caps add --uid=USER_ID--caps=CAPS
```

ユーザー、バケット、メタデータ、および使用状況 (使用率) に、読み取り、書き込み、またはすべての機能を追加できます。

構文

```
--caps="[users|buckets|metadata|usage|zone]=[*|read|write|read, write]"
```

例

```
[root@host01 ~]# radosgw-admin caps add --uid=johndoe --caps="users=*"
```

ユーザーから管理ケイパビリティーを削除するには、以下のコマンドを実行します。

例

```
[root@host01 ~]# radosgw-admin caps remove --uid=johndoe --caps={caps}
```

8.5. ロールの管理

ストレージ管理者は、**radosgw-admin** コマンドでロールに関連付けられたパーミッションを作成、削除、または更新できます。

ロールはユーザーに似ており、パーミッションポリシーが割り当てられます。任意のアイデンティティーをもとに想定することができます。ユーザーがロールを想定すると、動的に作成された一時認証情報のセットがユーザーに返されます。ロールを使用すると、一部の S3 リソースへのアクセス権限を持たないユーザー、アプリケーション、およびサービスへのアクセスを委譲できます。

8.5.1. ロールの作成

radosgw-admin role create コマンドでユーザーのロールを作成します。コマンドで **assume-role-policy-doc** パラメーターを持つユーザーを作成する必要があります。これは、エンティティーにロールを引き受けるパーミッションを付与する信頼関係ポリシードキュメントです。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- Ceph Object Gateway のインストール
- Ceph Object Gateway ノードへのルートレベルのアクセス。
- S3 バケットが作成されている。
- ユーザーアクセスで作成された S3 ユーザー。

手順

- ロールを作成します。

  構文

  ```
  radosgw-admin role create --role-name=ROLE_NAME [--path=="PATH_TO_FILE"] [--assume-role-policy-doc=TRUST_RELATIONSHIP_POLICY_DOCUMENT]
  ```

  例

  ```
  [root@host01 ~]# radosgw-admin role create --role-name=S3Access1 --path=/application_abc/component_xyz/ --assume-role-policy-doc=\{\"Version\":\"2012-10-17\",\"Statement\":\[\{\"Effect\":\"Allow\",\"Principal\":\{\"AWS\":\[\"arn:aws:iam:::user/TESTER\"\]\},\"Action\":\[\"sts:AssumeRole\"\]\}\]\}

  {
    "RoleId": "ca43045c-082c-491a-8af1-2eebca13deec",
    "RoleName": "S3Access1",
    "Path": "/application_abc/component_xyz/",
    "Arn": "arn:aws:iam:::role/application_abc/component_xyz/S3Access1",
    "CreateDate": "2022-06-17T10:18:29.116Z",
    "MaxSessionDuration": 3600,
    "AssumeRolePolicyDocument": "{\"Version\":\"2012-10-17\",\"Statement\":[{\"Effect\":\"Allow\",\"Principal\":{\"AWS\":[\"arn:aws:iam:::user/TESTER\"]},\"Action\":[\"sts:AssumeRole\"]}]}"
  }
  ```

  --path の値は、デフォルトで / です。

8.5.2. ロールの取得

get コマンドを使用して、ロールに関する情報を取得します。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- Ceph Object Gateway のインストール
- Ceph Object Gateway ノードへのルートレベルのアクセス。
- S3 バケットが作成されている。
- ロールが作成されている。
- ユーザーアクセスで作成された S3 ユーザー。

手順

- ロールに関する情報を取得します。

  構文

  ```
  radosgw-admin role get --role-name=ROLE_NAME
  ```

  例

  ```
  [root@host01 ~]# radosgw-admin role get --role-name=S3Access1

  {
    "RoleId": "ca43045c-082c-491a-8af1-2eebca13deec",
    "RoleName": "S3Access1",
    "Path": "/application_abc/component_xyz/",
    "Arn": "arn:aws:iam:::role/application_abc/component_xyz/S3Access1",
    "CreateDate": "2022-06-17T10:18:29.116Z",
    "MaxSessionDuration": 3600,
    "AssumeRolePolicyDocument": "{\"Version\":\"2012-10-17\",\"Statement\":
  [{\"Effect\":\"Allow\",\"Principal\":{\"AWS\":[\"arn:aws:iam:::user/TESTER\"]},\"Action\":
  [\"sts:AssumeRole\"]}]}"
  }
  ```

関連情報

- 詳細は、**Red Hat Ceph Storage Object Gateway ガイド**の **ロールの作成** セクションを参照してください。

8.5.3. ロールの一覧表示

list コマンドを使用して、特定のパス内のロールを一覧表示できます。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- Ceph Object Gateway のインストール

- Ceph Object Gateway ノードへのルートレベルのアクセス。
- S3 バケットが作成されている。
- ロールが作成されている。
- ユーザーアクセスで作成された S3 ユーザー。

手順

- ロールを一覧表示します。

  構文

  ```
  radosgw-admin role list
  ```

  例

  ```
  [root@host01 ~]# radosgw-admin role list

  [
     {
        "RoleId": "85fb46dd-a88a-4233-96f5-4fb54f4353f7",
        "RoleName": "kvm-sts",
        "Path": "/application_abc/component_xyz/",
        "Arn": "arn:aws:iam:::role/application_abc/component_xyz/kvm-sts",
        "CreateDate": "2022-09-13T11:55:09.39Z",
        "MaxSessionDuration": 7200,
        "AssumeRolePolicyDocument": "{\"Version\":\"2012-10-17\",\"Statement\":
  [{\"Effect\":\"Allow\",\"Principal\":{\"AWS\":[\"arn:aws:iam:::user/kvm\"]},\"Action\":
  [\"sts:AssumeRole\"]}]}"
     },
     {
        "RoleId": "9116218d-4e85-4413-b28d-cdfafba24794",
        "RoleName": "kvm-sts-1",
        "Path": "/application_abc/component_xyz/",
        "Arn": "arn:aws:iam:::role/application_abc/component_xyz/kvm-sts-1",
        "CreateDate": "2022-09-16T00:05:57.483Z",
        "MaxSessionDuration": 3600,
        "AssumeRolePolicyDocument": "{\"Version\":\"2012-10-17\",\"Statement\":
  [{\"Effect\":\"Allow\",\"Principal\":{\"AWS\":[\"arn:aws:iam:::user/kvm\"]},\"Action\":
  [\"sts:AssumeRole\"]}]}"
     }
  ]
  ```

8.5.4. ロールの仮定ロールポリシードキュメントの更新

modify コマンドを使用してロールを引き受けるためにエンティティーパーミッションを付与する assume ロールポリシードキュメントを更新できます。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。

- Ceph Object Gateway のインストール
- Ceph Object Gateway ノードへのルートレベルのアクセス。
- S3 バケットが作成されている。
- ロールが作成されている。
- ユーザーアクセスで作成された S3 ユーザー。

手順

- ロールの assume ロールポリシードキュメントを変更します。

  構文

  ```
  radosgw-admin role-trust-policy modify --role-name=ROLE_NAME --assume-role-policy-doc=TRUST_RELATIONSHIP_POLICY_DOCUMENT
  ```

  例

  ```
  [root@host01 ~]# radosgw-admin role-trust-policy modify --role-name=S3Access1 --assume-role-policy-doc=\{\"Version\":\"2012-10-17\",\"Statement\":\[\{\"Effect\":\"Allow\",\"Principal\":\{\"AWS\":\[\"arn:aws:iam:::user/TESTER\"\]\},\"Action\":\[\"sts:AssumeRole\"\]\}\]\}

  {
    "RoleId": "ca43045c-082c-491a-8af1-2eebca13deec",
    "RoleName": "S3Access1",
    "Path": "/application_abc/component_xyz/",
    "Arn": "arn:aws:iam:::role/application_abc/component_xyz/S3Access1",
    "CreateDate": "2022-06-17T10:18:29.116Z",
    "MaxSessionDuration": 3600,
    "AssumeRolePolicyDocument": "{\"Version\":\"2012-10-17\",\"Statement\":
  [{\"Effect\":\"Allow\",\"Principal\":{\"AWS\":[\"arn:aws:iam:::user/TESTER\"]},\"Action\":
  [\"sts:AssumeRole\"]}]}"
  }
  ```

8.5.5. ロールに割り当てられたパーミッションポリシーの取得

get コマンドを使用して、ロールに割り当てられた特定のパーミッションポリシーを取得できます。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- Ceph Object Gateway のインストール
- Ceph Object Gateway ノードへのルートレベルのアクセス。
- S3 バケットが作成されている。
- ロールが作成されている。
- ユーザーアクセスで作成された S3 ユーザー。

手順

- パーミッションポリシーを取得します。

  構文

  ```
  radosgw-admin role-policy get --role-name=ROLE_NAME --policy-name=POLICY_NAME
  ```

  例

  ```
  [root@host01 ~]# radosgw-admin role-policy get --role-name=S3Access1 --policy-name=Policy1

  {
    "Permission policy": "{\"Version\":\"2012-10-17\",\"Statement\":[{\"Effect\":\"Allow\",\"Action\":[\"s3:*\"],\"Resource\":\"arn:aws:s3:::example_bucket\"}]}"
  }
  ```

8.5.6. ロールの削除

ロールは、割り当てられたパーミッションポリシーを削除した後にのみ削除できます。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- Ceph Object Gateway のインストール
- Ceph Object Gateway ノードへのルートレベルのアクセス。
- ロールが作成されている。
- S3 バケットが作成されている。
- ユーザーアクセスで作成された S3 ユーザー。

手順

1. ロールに割り当てられたポリシーを削除します。

   構文

   ```
   radosgw-admin role policy delete --role-name=ROLE_NAME --policy-name=POLICY_NAME
   ```

   例

   ```
   [root@host01 ~]# radosgw-admin role policy delete --role-name=S3Access1 --policy-name=Policy1
   ```

2. ロールを削除します。

   構文

```
radosgw-admin role delete --role-name=ROLE_NAME
```

例

```
[root@host01 ~]# radosgw-admin role delete --role-name=S3Access1
```

8.5.7. ロールに割り当てられたポリシーの更新

put コマンドを使用して、ロールにアタッチされたインラインポリシーを追加または更新できます。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- Ceph Object Gateway のインストール
- Ceph Object Gateway ノードへのルートレベルのアクセス。
- S3 バケットが作成されている。
- ロールが作成されている。
- ユーザーアクセスで作成された S3 ユーザー。

手順

- インラインポリシーを更新します。

  構文

  ```
  radosgw-admin role-policy put --role-name=ROLE_NAME --policy-name=POLICY_NAME --
  policy-doc=PERMISSION_POLICY_DOCUMENT
  ```

  例

  ```
  [root@host01 ~]# radosgw-admin role-policy put --role-name=S3Access1 --policy-
  name=Policy1 --policy-doc=\{\"Version\":\"2012-10-17\",\"Statement\":\[\
  {\"Effect\":\"Allow\",\"Action\":\[\"s3:*\"\],\"Resource\":\"arn:aws:s3:::example_bucket\"\}\]\}
  ```

  この例では、**Policy1** をロール **S3Access1** に割り当てます。これにより、**example_bucket** ですべての S3 アクションが許可されます。

8.5.8. ロールに割り当てられたパーミッションポリシーの一覧表示

list コマンドを使用して、ロールに割り当てられているパーミッションポリシーの名前を一覧表示できます。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- Ceph Object Gateway のインストール

- Ceph Object Gateway ノードへのルートレベルのアクセス。
- S3 バケットが作成されている。
- ロールが作成されている。
- ユーザーアクセスで作成された S3 ユーザー。

手順

- パーミッションポリシーの名前を一覧表示します。

  構文

  ```
  radosgw-admin role-policy list --role-name=ROLE_NAME
  ```

  例

  ```
  [root@host01 ~]# radosgw-admin role-policy list --role-name=S3Access1

  [
    "Policy1"
  ]
  ```

8.5.9. ロールに割り当てられたポリシーの削除

rm コマンドを使用すると、ロールに割り当てられたパーミッションポリシーを削除できます。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- Ceph Object Gateway のインストール
- Ceph Object Gateway ノードへのルートレベルのアクセス。
- S3 バケットが作成されている。
- ロールが作成されている。
- ユーザーアクセスで作成された S3 ユーザー。

手順

- パーミッションポリシーを削除します。

  構文

  ```
  radosgw-admin role policy delete --role-name=ROLE_NAME --policy-name=POLICY_NAME
  ```

  例

```
[root@host01 ~]# radosgw-admin role policy delete --role-name=S3Access1 --policy-
name=Policy1
```

8.5.10. ロールのセッション期間の更新

update コマンドを使用してロールのセッション期間を更新し、提供された認証情報を使用してユーザーがアカウントにサインインできる期間を制御できます。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- Ceph Object Gateway のインストール
- Ceph Object Gateway ノードへのルートレベルのアクセス。
- S3 バケットが作成されている。
- ロールが作成されている。
- ユーザーアクセスで作成された S3 ユーザー。

手順

- **update** コマンドを使用して max-session-duration を更新します。

  構文

  ```
  [root@node1 ~]# radosgw-admin role update --role-name=ROLE_NAME --max-session-
  duration=7200
  ```

  例

  ```
  [root@node1 ~]# radosgw-admin role update --role-name=test-sts-role --max-session-
  duration=7200
  ```

検証

- ロールを一覧表示して、更新を確認します。

  例

  ```
  [root@node1 ~]#radosgw-admin role list
  [
     {
        "RoleId": "d4caf33f-caba-42f3-8bd4-48c84b4ea4d3",
        "RoleName": "test-sts-role",
        "Path": "/",
        "Arn": "arn:aws:iam:::role/test-role",
        "CreateDate": "2022-09-07T20:01:15.563Z",
        "MaxSessionDuration": 7200,    <<<<<<
        "AssumeRolePolicyDocument": "{\"Version\":\"2012-10-17\",\"Statement\":
  [{\"Effect\":\"Allow\",\"Principal\":{\"AWS\":[\"arn:aws:iam:::user/kvm\"]},\"Action\":
  ```

```
[\"sts:AssumeRole\"]}]}"
   }
]
```

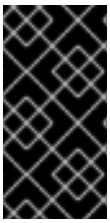
関連情報

- 詳細は、**Red Hat Ceph Storage 開発者ガイド**の **ロールを操作する REST API** セクションを参照してください。

8.6. クォータ管理

Ceph Object Gateway を使用すると、ユーザーが所有するユーザーおよびバケットにクォータを設定することができます。クォータには、バケットのオブジェクトの最大数と、メガバイト単位のストレージの最大サイズが含まれます。

- Bucket: **--bucket** オプションでは、ユーザーが所有するバケットのクォータを指定できます。
- Maximum Objects: **--max-objects** 設定では、オブジェクトの最大数を指定できます。負の値を設定すると、この設定が無効になります。
- Maximum Size: **--max-size** オプションでは、バイトの最大数のクォータを指定できます。負の値を設定すると、この設定が無効になります。
- Quota Scope: **--quota-scope** オプションは、クォータのスコープを設定します。オプションは **bucket** と **user** です。バケットクォータは、ユーザーが所有するバケットに適用されます。ユーザークォータはユーザーに適用されます。



重要

多数のオブジェクトを含むバケットは、深刻なパフォーマンスの問題を引き起こす可能性があります。1 つのバケット内のオブジェクトの推奨最大数は 100,000 です。この数を増やすには、バケットインデックスシャーディングを設定します。詳細は、**バケットインデックスの再シャーディングの設定** を参照してください。

8.6.1. ユーザークォータの設定

クォータを有効にする前に、まずクォータパラメーターを設定する必要があります。

構文

```
radosgw-admin quota set --quota-scope=user --uid=USER_ID [--max-objects=NUMBER_OF_OBJECTS] [--max-size=MAXIMUM_SIZE_IN_BYTES]
```

例

```
[root@host01 ~]# radosgw-admin quota set --quota-scope=user --uid=johndoe --max-objects=1024 --max-size=1024
```

num オブジェクトおよび/または最大サイズの負の値は、特定のクォータ属性チェックが無効になっていることを意味します。

8.6.2. ユーザークォータの有効化と無効化

ユーザークォータを設定したら、これを有効にすることができます。

構文

```
radosgw-admin quota enable --quota-scope=user --uid=USER_ID
```

有効なユーザークォータを無効にすることができます。

構文

```
radosgw-admin quota disable --quota-scope=user --uid=USER_ID
```

8.6.3. バケットクォータの設定

バケットクォータは、指定された **uid** が所有するバケットに適用されます。ユーザーからは独立しています。

構文

```
radosgw-admin quota set --uid=USER_ID --quota-scope=bucket --bucket=BUCKET_NAME [--max-objects=NUMBER_OF_OBJECTS] [--max-size=MAXIMUM_SIZE_IN_BYTES]
```

NUMBER_OF_OBJECTS、MAXIMUM_SIZE_IN_BYTES、またはその両方が負の値の場合、特定のクォータ属性チェックが無効になっていることを意味します。

8.6.4. バケットクォータの有効化と無効化

バケットクォータを設定したら、これを有効にすることができます。

構文

```
radosgw-admin quota enable --quota-scope=bucket --uid=USER_ID
```

有効なバケットクォータを無効にすることができます。

構文

```
radosgw-admin quota disable --quota-scope=bucket --uid=USER_ID
```

8.6.5. クォータ設定の取得

ユーザー情報 API を使用して、各ユーザーのクォータ設定にアクセスすることができます。CLI インターフェイスでユーザークォータ設定情報を読み取るには、以下のコマンドを実行します。

構文

```
radosgw-admin user info --uid=USER_ID
```

テナントユーザーのクォータ設定を取得するには、ユーザー ID とテナントの名前を指定します。

構文

-

```
radosgw-admin user info --uid=USER_ID --tenant=TENANT
```

8.6.6. クォータ統計の更新

クォータ統計は非同期で更新されます。すべてのユーザーおよびすべてのバケットのクォータ統計を手動で更新して、最新のクォータ統計を取得できます。

構文

```
radosgw-admin user stats --uid=USER_ID --sync-stats
```

8.6.7. ユーザークォータ使用の統計の取得

ユーザーが消費したクォータの量を確認するには、以下のコマンドを実行します。

構文

```
radosgw-admin user stats --uid=USER_ID
```

注記

最新のデータを取得するには、**--sync-stats** オプションを指定して **radosgw-admin user stats** コマンドを実行する必要があります。

8.6.8. クォータキャッシュ

クォータ統計は、各 Ceph Gateway インスタンスに対してキャッシュされます。複数のインスタンスがある場合、インスタンスごとにクォータのビューが異なるため、キャッシュによってクォータが完全に適用されないようにすることができます。これを制御するオプションは、**rgw bucket quota ttl**、**rgw user quota bucket sync interval**、および **rgw user quota sync interval** です。これらの値が高いほど、クォータ操作は効率的ですが、複数のインスタンスが同期しなくなります。これらの値が低いほど、複数のインスタンスは完全に近い形で適用されます。3 つすべてが 0 の場合には、クォータキャッシュは実質的に無効になり、複数のインスタンスで完全にクォータが適用されます。

8.6.9. グローバルクォータの読み取りおよび作成

ゾーングループマップでクォータ設定を読み書きできます。ゾーングループマップを取得するには、次のコマンドを実行します。

```
[root@host01 ~]# radosgw-admin global quota get
```

グローバルクォータ設定は、**quota set**、**quota enable**、および **quota disable** コマンドに対応する **global quota** で操作できます。次に例を示します。

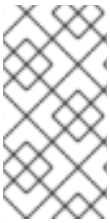
```
[root@host01 ~]# radosgw-admin global quota set --quota-scope bucket --max-objects 1024
[root@host01 ~]# radosgw-admin global quota enable --quota-scope bucket
```

注記

レルムと期間が存在するマルチサイト設定では、グローバルクォータへの変更は、**period update --commit** を使用してコミットする必要があります。期間が表示されていない場合、変更を有効にするには、Ceph Object Gateway を再起動する必要があります。

8.7. バケット管理

ストレージ管理者は、Ceph Object Gateway を使用する場合は、バケットをユーザー間で移動して名前を変更することで、バケットを管理できます。バケット通知を作成して、特定のイベントでトリガーできます。また、ストレージクラスターの存続期間中に発生する可能性のある孤立したオブジェクトやリークオブジェクトを Ceph Object Gateway 内で見つけることができます。



注記

何百万ものオブジェクトが高い取り込み率で Ceph Object Gateway バケットにアップロードされると、**radosgw-admin bucket stats** コマンドで誤った **num_objects** が報告されます。**radosgw-admin bucket list** コマンドを使用すると、**num_objects** パラメーターの値を修正できます。

注記

マルチサイトクラスターでは、セカンダリーサイトからバケットを削除しても、メタデータの変更はプライマリーサイトと同期されません。したがって、Red Hat では、セカンダリーサイトではなく、プライマリーサイトからのみバケットを削除することを推奨します。

8.7.1. バケットの名前変更

バケットの名前を変更できます。バケット名のアンダースコアを許可する必要がある場合は、**rgw_relaxed_s3_bucket_names** オプションを **true** に設定します。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- Ceph Object Gateway ソフトウェアのインストール。
- 既存のバケット。

手順

1. バケットをリスト表示します。

   例

```
[ceph: root@host01 /]# radosgw-admin bucket list
[
    "34150b2e9174475db8e191c188e920f6/swcontainer",
    "s3bucket1",
    "34150b2e9174475db8e191c188e920f6/swimpfalse",
    "c278edd68cfb4705bb3e07837c7ad1a8/ec2container",
    "c278edd68cfb4705bb3e07837c7ad1a8/demoten1",
```

```
    "c278edd68cfb4705bb3e07837c7ad1a8/demo-ct",
    "c278edd68cfb4705bb3e07837c7ad1a8/demopostup",
    "34150b2e9174475db8e191c188e920f6/postimpfalse",
    "c278edd68cfb4705bb3e07837c7ad1a8/demoten2",
    "c278edd68cfb4705bb3e07837c7ad1a8/postupsw"
]
```

2. バケットの名前を変更します。

構文

```
radosgw-admin bucket link --bucket=ORIGINAL_NAME --bucket-new-name=NEW_NAME --uid=USER_ID
```

例

```
[ceph: root@host01 /]# radosgw-admin bucket link --bucket=s3bucket1 --bucket-new-name=s3newb --uid=testuser
```

バケットがテナント内部にある場合は、テナントも指定します。

構文

```
radosgw-admin bucket link --bucket=tenant/ORIGINAL_NAME --bucket-new-name=NEW_NAME --uid=TENANT$USER_ID
```

例

```
[ceph: root@host01 /]# radosgw-admin bucket link --bucket=test/s3bucket1 --bucket-new-name=s3newb --uid=test$testuser
```

3. バケットの名前が変更されたことを確認します。

例

```
[ceph: root@host01 /]# radosgw-admin bucket list
[
    "34150b2e9174475db8e191c188e920f6/swcontainer",
    "34150b2e9174475db8e191c188e920f6/swimpfalse",
    "c278edd68cfb4705bb3e07837c7ad1a8/ec2container",
    "s3newb",
    "c278edd68cfb4705bb3e07837c7ad1a8/demoten1",
    "c278edd68cfb4705bb3e07837c7ad1a8/demo-ct",
    "c278edd68cfb4705bb3e07837c7ad1a8/demopostup",
    "34150b2e9174475db8e191c188e920f6/postimpfalse",
    "c278edd68cfb4705bb3e07837c7ad1a8/demoten2",
    "c278edd68cfb4705bb3e07837c7ad1a8/postupsw"
]
```

8.7.2. バケットの削除

Ceph Object Gateway 設定を使用して Red Hat Ceph Storage クラスターからバケットを削除します。

バケットにオブジェクトがない場合は、**radosgw-admin bucket rm** コマンドを実行できます。バケット内にオブジェクトがある場合は、**--purge-objects** オプションを使用できます。

マルチサイト設定の場合、Red Hat はプライマリーサイトからバケットを削除することを推奨します。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- Ceph Object Gateway ソフトウェアのインストール。
- 既存のバケット。

手順

1. バケットをリスト表示します。

   例

   ```
   [ceph: root@host01 /]# radosgw-admin bucket list
   [
       "34150b2e9174475db8e191c188e920f6/swcontainer",
       "s3bucket1",
       "34150b2e9174475db8e191c188e920f6/swimpfalse",
       "c278edd68cfb4705bb3e07837c7ad1a8/ec2container",
       "c278edd68cfb4705bb3e07837c7ad1a8/demoten1",
       "c278edd68cfb4705bb3e07837c7ad1a8/demo-ct",
       "c278edd68cfb4705bb3e07837c7ad1a8/demopostup",
       "34150b2e9174475db8e191c188e920f6/postimpfalse",
       "c278edd68cfb4705bb3e07837c7ad1a8/demoten2",
       "c278edd68cfb4705bb3e07837c7ad1a8/postupsw"
   ]
   ```

2. バケットを削除します。

   構文

   ```
   radosgw-admin bucket rm --bucket=BUCKET_NAME
   ```

   例

   ```
   [ceph: root@host01 /]# radosgw-admin bucket rm --bucket=s3bucket1
   ```

3. バケットにオブジェクトがある場合は、次のコマンドを実行します。

   構文

   ```
   radosgw-admin bucket rm --bucket=BUCKET --purge-objects --bypass-gc
   ```

   例

   ```
   [ceph: root@host01 /]# radosgw-admin bucket rm --bucket=s3bucket1 --purge-objects --
   bypass-gc
   ```

--purge-objects オプションはオブジェクトをパージし、**--bypass-gc** オプションはガベージコレクターを使用せずにオブジェクトの削除をトリガーして、プロセスをより効率的にします。

4. バケットが削除されたことを確認します。

例

```
[ceph: root@host01 /]# radosgw-admin bucket list
[
    "34150b2e9174475db8e191c188e920f6/swcontainer",
    "34150b2e9174475db8e191c188e920f6/swimpfalse",
    "c278edd68cfb4705bb3e07837c7ad1a8/ec2container",
    "c278edd68cfb4705bb3e07837c7ad1a8/demoten1",
    "c278edd68cfb4705bb3e07837c7ad1a8/demo-ct",
    "c278edd68cfb4705bb3e07837c7ad1a8/demopostup",
    "34150b2e9174475db8e191c188e920f6/postimpfalse",
    "c278edd68cfb4705bb3e07837c7ad1a8/demoten2",
    "c278edd68cfb4705bb3e07837c7ad1a8/postupsw"
]
```

8.7.3. バケットの移動

radosgw-admin bucket ユーティリティーは、ユーザー間でバケットを移行する機能を提供します。これを実行するには、バケットを新規ユーザーにリンクし、バケットの所有権を新規ユーザーに変更します。

バケットを移動できます。

- テナントのない 2 人のユーザー間
- 2 人のテナントユーザー間
- テナントのないユーザーとテナントユーザーとの間

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- Ceph Object Gateway がインストールされている。
- S3 バケット。
- さまざまなテナントユーザーとテナントのないユーザー。

8.7.3.1. テナントのないユーザー間でのバケットの移動

radosgw-admin bucket chown コマンドは、バケットとそれに含まれるすべてのオブジェクトの所有権をあるユーザーから別のユーザーに変更する機能を提供します。これを行うには、バケットを現在のユーザーからリンク解除し、新しいユーザーにリンクして、バケットの所有権を新しいユーザーに変更します。

手順

1. バケットを新規ユーザーにリンクします。

構文

```
radosgw-admin bucket link --uid=USER --bucket=BUCKET
```

例

```
[ceph: root@host01 /]# radosgw-admin bucket link --uid=user2 --bucket=data
```

2. バケットが **user2** に正常にリンクされていることを確認します。

例

```
[ceph: root@host01 /]# radosgw-admin bucket list --uid=user2
[
    "data"
]
```

3. バケットの所有権を新規ユーザーに変更します。

構文

```
radosgw-admin bucket chown --uid=user --bucket=bucket
```

例

```
[ceph: root@host01 /]# radosgw-admin bucket chown --uid=user2 --bucket=data
```

4. 次のコマンドの出力で **owner** 行を確認して、**data** バケットの所有権が正常に変更されたことを確認します。

例

```
[ceph: root@host01 /]# radosgw-admin bucket list --bucket=data
```

8.7.3.2. テナントユーザー間でのバケットの移動

バケットは、あるテナントユーザーと別のテナントユーザーの間を移動できます。

手順

1. バケットを新規ユーザーにリンクします。

構文

```
radosgw-admin bucket link --bucket=CURRENT_TENANT/BUCKET --
uid=NEW_TENANT$USER
```

例

```
[ceph: root@host01 /]# radosgw-admin bucket link --bucket=test/data --uid=test2$user2
```

2. バケットが **user2** に正常にリンクされていることを確認します。

```
[ceph: root@host01 /]# radosgw-admin bucket list --uid=test$user2
[
    "data"
]
```

3. バケットの所有権を新規ユーザーに変更します。

構文

```
radosgw-admin bucket chown --bucket=NEW_TENANT/BUCKET --
uid=NEW_TENANT$USER
```

例

```
[ceph: root@host01 /]# radosgw-admin bucket chown --bucket='test2/data' --uid='test$tuser2'
```

4. 次のコマンドの出力で **owner** 行を確認して、**data** バケットの所有権が正常に変更されたことを確認します。

```
[ceph: root@host01 /]# radosgw-admin bucket list --bucket=test2/data
```

8.7.3.3. バケットをテナントのないユーザーからテナントユーザーに移動する

バケットをテナントのないユーザーからテナントユーザーに移動できます。

手順

1. オプション: まだ複数のテナントがない場合は、**rgw_keystone_implicit_tenants** を有効にして、外部テナントから Ceph Object Gateway にアクセスすることでテナントを作成できます。**rgw_keystone_implicit_tenants** オプションを有効にします。

例

```
[ceph: root@host01 /]# ceph config set client.rgw rgw_keystone_implicit_tenants true
```

s3cmd コマンドまたは **swift** コマンドのいずれかを使用して、一時テナントから Ceph Object Gateway にアクセスします。

例

```
[ceph: root@host01 /]# swift list
```

または、**s3cmd** を使用します。

例

```
[ceph: root@host01 /]# s3cmd ls
```

外部テナントからの最初のアクセスにより、同等の Ceph Object Gateway ユーザーが作成されます。

2. バケットをテナントされたユーザーに移動します。

   構文

   ```
   radosgw-admin bucket link --bucket=/BUCKET --uid='TENANT$USER'
   ```

   例

   ```
   [ceph: root@host01 /]# radosgw-admin bucket link --bucket=/data --uid='test$tenanted-user'
   ```

3. **data** バケットが **tenanted-user** に正常にリンクされていることを確認します。

   例

   ```
   [ceph: root@host01 /]# radosgw-admin bucket list --uid='test$tenanted-user'
   [
       "data"
   ]
   ```

4. バケットの所有権を新規ユーザーに変更します。

   構文

   ```
   radosgw-admin bucket chown --bucket='tenant/bucket name' --uid='tenant$user'
   ```

   例

   ```
   [ceph: root@host01 /]# radosgw-admin bucket chown --bucket='test/data' --
   uid='test$tenanted-user'
   ```

5. 次のコマンドの出力で **owner** 行を確認して、**data** バケットの所有権が正常に変更されたことを確認します。

   例

   ```
   [ceph: root@host01 /]# radosgw-admin bucket list --bucket=test/data
   ```

8.7.3.4. 孤立したオブジェクトやリークオブジェクトを見つける

正常なストレージクラスターには孤立したオブジェクトやリークオブジェクトがありませんが、場合によっては、孤立したオブジェクトやリークオブジェクトが発生する可能性があります。

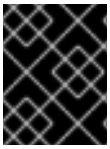
孤立オブジェクトはストレージクラスター内に存在し、RADOS オブジェクトに関連付けられたオブジェクト ID を持ちます。ただし、バケットインデックス参照には、S3 オブジェクトを含む RADOS オブジェクトの参照がありません。たとえば、Ceph Object Gateway が操作の途中でダウンした場合、一部のオブジェクトが孤立する原因となる可能性があります。また、検出されないバグでも、孤立したオブジェクトが発生する可能性があります。

Ceph Object Gateway オブジェクトが RADOS オブジェクトにどのようにマッピングされるかを確認することができます。**radosgw-admin** コマンドは、これらの潜在的な孤立オブジェクトまたはリークオブジェクトのリストを検索して生成するための新しいツールを提供します。**radoslist** サブコマンドを

使用すると、バケット内に保存されているオブジェクト、またはストレージクラスター内のすべてのバケットが表示されます。**rgw-orphan-list** スクリプトは、プール内の孤立したオブジェクトを表示します。

注記

radoslist サブコマンドは、非推奨の **orphans find** サブコマンドおよび **orphans finish** サブコマンドを置き換えます。



重要

すべてのオブジェクトが **orphaned** として表示されるため、**Indexless** バケットが使用されている場合は、このコマンドを使用しないでください。

孤立したオブジェクトを識別するもう 1 つの代替方法として、**rados -p <pool> ls | grep BUCKET_ID** コマンドを実行する方法があります。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- 実行中の Ceph Object Gateway。

手順

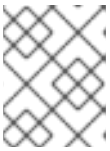
1. バケット内でデータを保持するオブジェクトのリストを生成するには、以下を実行します。

   構文

   ```
   radosgw-admin bucket radoslist --bucket BUCKET_NAME
   ```

   例

   ```
   [root@host01 ~]# radosgw-admin bucket radoslist --bucket mybucket
   ```

   注記

   BUCKET_NAME を省略すると、すべてのバケット内のすべてのオブジェクトが表示されます。

2. **rgw-orphan-list** のバージョンを確認します。

   例

   ```
   [root@host01 ~]# head /usr/bin/rgw-orphan-list
   ```

   バージョンは **2023-01-11** 以降である必要があります。

3. orphan のリストを生成する必要があるディレクトリーを作成します。

   例

```
[root@host01 ~]# mkdir orphans
```

4. 前に作成したディレクトリーに移動します。

例

```
[root@host01 ~]# cd orphans
```

5. プールリストから、orphans を検索するプールを選択します。このスクリプトは、クラスター内のオブジェクトによっては長時間実行される可能性があります。

例

```
[root@host01 orphans]# rgw-orphan-list
```

例

```
Available pools:
   .rgw.root
   default.rgw.control
   default.rgw.meta
   default.rgw.log
   default.rgw.buckets.index
   default.rgw.buckets.data
   rbd
   default.rgw.buckets.non-ec
   ma.rgw.control
   ma.rgw.meta
   ma.rgw.log
   ma.rgw.buckets.index
   ma.rgw.buckets.data
   ma.rgw.buckets.non-ec
Which pool do you want to search for orphans?
```

プール名を入力して、孤立を検索します。

重要

メタデータプールではなく、**rgw-orphan-list** コマンドを使用する場合は、データプールを指定する必要があります。

6. **rgw-orphan-list** ツールの使用の詳細を表示します。

構文

```
rgw-orphan-list -h
rgw-orphan-list POOL_NAME /DIRECTORY
```

例

```
[root@host01 orphans]# rgw-orphan-list default.rgw.buckets.data /orphans
```

```
2023-09-12 08:41:14 ceph-host01 Computing delta...
2023-09-12 08:41:14 ceph-host01 Computing results...
10 potential orphans found out of a possible 2412 (0%).          <<<<<<< orphans detected
The results can be found in './orphan-list-20230912124113.out'.
    Intermediate files are './rados-20230912124113.intermediate' and './radosgw-admin-
20230912124113.intermediate'.
***
*** WARNING: This is EXPERIMENTAL code and the results should be used
***          only with CAUTION!
***
Done at 2023-09-12 08:41:14.
```

7. **ls -l** コマンドを実行して、エラーで終わるファイルの長さが 0 であることを確認し、スクリプトが問題なく実行されたことを示します。

   例

```
[root@host01 orphans]# ls -l

-rw-r--r--. 1 root root    770 Sep 12 03:59 orphan-list-20230912075939.out
-rw-r--r--. 1 root root      0 Sep 12 03:59 rados-20230912075939.error
-rw-r--r--. 1 root root 248508 Sep 12 03:59 rados-20230912075939.intermediate
-rw-r--r--. 1 root root      0 Sep 12 03:59 rados-20230912075939.issues
-rw-r--r--. 1 root root      0 Sep 12 03:59 radosgw-admin-20230912075939.error
-rw-r--r--. 1 root root 247738 Sep 12 03:59 radosgw-admin-20230912075939.intermediate
```

8. リストされた孤立オブジェクトを確認します。

   例

```
[root@host01 orphans]# cat ./orphan-list-20230912124113.out

a9c042bc-be24-412c-9052-dda6b2f01f55.16749.1_key1.cherylf.433-bucky-4865-0.0
a9c042bc-be24-412c-9052-dda6b2f01f55.16749.1_key1.cherylf.433-bucky-4865-0.1
a9c042bc-be24-412c-9052-dda6b2f01f55.16749.1_key1.cherylf.433-bucky-4865-0.2
a9c042bc-be24-412c-9052-dda6b2f01f55.16749.1_key1.cherylf.433-bucky-4865-0.3
a9c042bc-be24-412c-9052-dda6b2f01f55.16749.1_key1.cherylf.433-bucky-4865-0.4
a9c042bc-be24-412c-9052-dda6b2f01f55.16749.1_key1.cherylf.433-bucky-4865-0.5
a9c042bc-be24-412c-9052-dda6b2f01f55.16749.1_key1.cherylf.433-bucky-4865-0.6
a9c042bc-be24-412c-9052-dda6b2f01f55.16749.1_key1.cherylf.433-bucky-4865-0.7
a9c042bc-be24-412c-9052-dda6b2f01f55.16749.1_key1.cherylf.433-bucky-4865-0.8
a9c042bc-be24-412c-9052-dda6b2f01f55.16749.1_key1.cherylf.433-bucky-4865-0.9
```

9. 孤立したオブジェクトを削除します。

   構文

```
rados -p POOL_NAME rm OBJECT_NAME
```

   例

```
[root@host01 orphans]# rados -p default.rgw.buckets.data rm myobject
```

警告

正しいオブジェクトを削除していることを確認してください。**rados rm** コマンドを実行すると、ストレージクラスターからデータが削除されます。

8.7.3.5. バケットインデックスエントリーの管理

radosgw-admin bucket check サブコマンドを使用して、Red Hat Ceph Storage クラスターで Ceph Object Gateway のバケットインデックスエントリーを管理できます。

マルチパートアップロードオブジェクトの一部に関連する各バケットインデックスエントリーは、対応する **.meta** インデックスエントリーと照合されます。特定のマルチパートアップロードのすべての部分に **.meta** エントリーが必要です。ピースに対応する **.meta** エントリーが見つからない場合、出力のセクションに孤立したエントリーが一覧表示されます。

バケットの統計はバケットインデックスヘッダーに保存されます。このフェーズでは、これらのヘッダーをロードし、バケットインデックスのすべてのプレーンオブジェクトエントリーを繰り返し処理し、統計を再計算します。次に、それぞれ existing_header と calculated_header というラベルの付いたセクションに実際の統計と計算した統計を表示して、比較できるようにします。

バケットチェック サブコマンドで **--fix** オプションを使用すると、孤立したエントリーがバケットインデックスから削除され、ヘッダー内の既存の統計が計算された統計で上書きされます。これにより、バージョン管理で使用される複数のエントリーを含むすべてのエントリーが出力に一覧表示されます。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- 実行中の Ceph Object Gateway。
- 新規に作成されたバケット。

手順

1. 特定のバケットのバケットインデックスを確認します。

   構文

   ```
   radosgw-admin bucket check --bucket=BUCKET_NAME
   ```

   例

   ```
   [root@rgw ~]# radosgw-admin bucket check --bucket=mybucket
   ```

2. 孤立したオブジェクトの削除など、バケットインデックスの不整合を修正します。

   構文

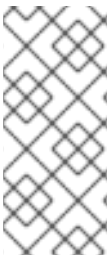
   ```
   radosgw-admin bucket check --fix --bucket=BUCKET_NAME
   ```

例

```
[root@rgw ~]# radosgw-admin bucket check --fix --bucket=mybucket
```

8.7.3.6. バケット通知

バケット通知により、バケットで特定のイベントが発生した場合に、Ceph Object Gateway から情報を送る方法が提供されます。バケット通知は HTTP、AMQP0.9.1、および Kafka エンドポイントに送信できます。特定バケットおよび特定のトピック上のイベントのバケット通知を送信するために、通知エントリーを作成する必要があります。バケット通知は、イベントタイプのサブセットに作成することも、デフォルトですべてのイベントタイプに対して作成できます。バケット通知は、キーの接頭辞または接尾辞、キーに一致する正規表現、オブジェクトに割り当てられたメタデータ属性、またはオブジェクトタグに基づいてイベントをフィルタリングできます。バケット通知には、バケット通知メカニズムの設定および制御インターフェイスを提供する REST API があります。



注記

バケット通知 API はデフォルトで有効にされます。**rgw_enable_apis** 設定パラメーターを明示的に設定する場合は、**s3** および **notifications** が含まれていることを確認してください。これを確認するには、**ceph --admin-daemon /var/run/ceph/ceph-client.rgw.NAME.asok config get rgw_enable_apis** コマンドを実行します。**NAME** を、Ceph Object Gateway インスタンス名に置き換えます。

CLI を使用したトピック管理

Ceph Object Gateway バケットのトピックのリスト表示、取得、および削除を実行できます。

- **トピックのリスト表**: 以下のコマンドを実行し、すべてのトピックの設定をリスト表示します。

  例

  ```
  [ceph: host01 /]# radosgw-admin topic list
  ```

- **トピックの取得**: 以下のコマンドを実行して、特定のトピックの設定を取得します。

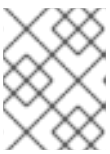
  例

  ```
  [ceph: host01 /]# radosgw-admin topic get --topic=topic1
  ```

- **トピックの削除**: 以下のコマンドを実行し、特定のトピックの設定を削除します。

  例

  ```
  [ceph: host01 /]# radosgw-admin topic rm --topic=topic1
  ```

  注記

  Ceph Object Gateway バケットがそのトピックに設定されている場合でも、トピックが削除されます。

8.7.3.7. バケット通知の作成

バケットレベルでバケット通知を作成します。通知設定には、Red Hat Ceph Storage Object Gateway S3 イベント、**ObjectCreated**、**ObjectRemoved**、および **ObjectLifecycle:Expiration** があります。これらは、バケット通知を送信するために宛先とともに公開する必要があります。バケット通知は S3 オペレーションです。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- 稼働中の HTTP サーバー、RabbitMQ サーバー、または Kafka サーバー。
- ルートレベルのアクセス。
- Red Hat Ceph Storage Object Gateway のインストール
- ユーザーアクセスキーおよびシークレットキー。
- エンドポイントパラメーター。

重要

Red Hat は、**ObjectCreate** イベント (例: **put**、**post**、**multipartUpload**、および **copy**) をサポートします。また、Red Hat は、**object_delete**、**s3_multi_object_delete** などの **ObjectRemove** イベントをサポートしています。

手順

1. S3 バケットを作成します。
2. **http**、**amqp**、または **kafka** プロトコルに SNS トピックを作成します。
3. **s3:objectCreate**、**s3:objectRemove**、および **s3:ObjectLifecycle:Expiration** イベントの S3 バケット通知を作成します。

   例

```
client.put_bucket_notification_configuration(
   Bucket=bucket_name,
   NotificationConfiguration={
      'TopicConfigurations': [
         {
            'Id': notification_name,
            'TopicArn': topic_arn,
            'Events': ['s3:ObjectCreated:*', 's3:ObjectRemoved:*',
's3:ObjectLifecycle:Expiration:*']
         }]})
```

4. バケットに S3 オブジェクトを作成します。
5. レシーバー **http**、**rabbitmq**、または **kafka** でのオブジェクト作成イベントを確認します。
6. オブジェクトを削除します。
7. レシーバー **http**、**rabbitmq**、または **kafka** でオブジェクトの削除イベントを確認します。

8.7.4. S3 バケットレプリケーション API (テクノロジープレビュー)

重要

S3 バケットレプリケーションは、Red Hat Ceph Storage 7.0 のみで利用可能なテクノロジープレビュー機能です。テクノロジープレビュー機能は、実稼働環境での Red Hat サービスレベルアグリーメント (SLA) ではサポートされておらず、機能的に完全ではない可能性があるため、Red Hat では実稼働環境での使用を推奨していません。テクノロジープレビューの機能は、最新の製品機能をいち早く提供して、開発段階で機能のテストを行いフィードバックを提供していただくことを目的としています。

詳細は、Red Hat テクノロジープレビュー機能のサポート範囲 を参照してください。

S3 バケットレプリケーション API も実装され、ユーザーは異なるバケット間でレプリケーションルールを作成できるようになりました。AWS レプリケーション機能により、同じゾーン内のバケットレプリケーションが許可されますが、現時点では Ceph Object Gateway は許可しません。ただし、Ceph Object Gateway API には、特定のバケットを同期するゾーンをユーザーが選択できる **Zone** 配列も追加されました。

8.7.4.1. S3 バケットレプリケーションの作成

バケットのレプリケーション設定を作成するか、既存のレプリケーション設定を置き換えます。

レプリケーション設定には少なくとも 1 つのルールが含まれている必要があります。各ルールは、ソースバケット内のオブジェクトをフィルタリングすることにより、レプリケートするオブジェクトのサブセットを識別します。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- Ceph Object Gateway がインストールされている。

手順

1. レプリケーションの詳細を含むレプリケーション設定ファイルを作成します。

   構文

```
{
"Role": "arn:aws:iam::account-id:role/role-name",
   "Rules": [
      {
         "Status": "Enabled",
         "Priority": 1,
         "DeleteMarkerReplication": { "Status": "Disabled" },
         "Filter" : { "Prefix": ""},
         "Destination": {
            "Bucket": "BUCKET_NAME"
         }
      }
   ]
}
```

例

```
[root@host01 ~]# cat replication.json
{
   "Role": "arn:aws:iam::account-id:role/role-name",
   "Rules": [
      {
         "Status": "Enabled",
         "Priority": 1,
         "DeleteMarkerReplication": { "Status": "Disabled" },
         "Filter" : { "Prefix": ""},
         "Destination": {
            "Bucket": "testbucket"
         }
      }
   ]
}
```

2. S3 API の put バケットレプリケーションを作成します。

構文

```
aws --endpoint-url=RADOSGW_ENDPOINT_URL s3api put-bucket-replication --bucket
BUCKET_NAME --replication-configuration file://REPLICATION_CONFIIRATION_FILE.json
```

例

```
[root@host01 ~]# aws --endpoint-url=http://host01:80 s3api put-bucket-replication --bucket
testbucket --replication-configuration file://replication.json
```

8.7.4.2. S3 バケットのレプリケーションを取得する

バケットのレプリケーション設定を取得できます。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- Ceph Object Gateway がインストールされている。
- S3 バケットのレプリケーションが作成されている。

手順

- S3 API の put バケットレプリケーションを取得します。

構文

```
aws s3api get-bucket-replication --bucket BUCKET_NAME --endpoint-
url=RADOSGW_ENDPOINT_URL
```

例

```
[root@host01 ~]# aws s3api get-bucket-replication --bucket testbucket --endpoint-
url=http://host01:80
{

"ReplicationConfiguration": {
    "Role": "arn:aws:iam::account-id:role/role-name",
    "Rules": [
        {
            "Status": "Enabled",
            "Priority": 1,
            "DeleteMarkerReplication": { "Status": "Disabled" },
            "Filter" : { "Prefix": ""},
            "Destination": {
                Bucket": "testbucket"
            }
        }
    ]
 }

}
```

8.7.4.3. S3 バケットレプリケーションの削除

バケットからレプリケーション設定を削除します。

バケット所有者は、レプリケーション設定を削除する権限を他のユーザーに付与できます。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- Ceph Object Gateway がインストールされている。
- S3 バケットのレプリケーションが作成されている。

手順

1. S3 API の put バケットレプリケーションを削除します。

   構文

   ```
   aws s3api delete-bucket-replication --bucket BUCKET_NAME --endpoint-
   url=RADOSGW_ENDPOINT_URL
   ```

   例

   ```
   [root@host01 ~]# aws s3api delete-bucket-replication --bucket testbucket --endpoint-
   url=http://host01:80
   ```

2. 既存のレプリケーションルールが削除されていることを確認します。

   構文

```
aws s3api get-bucket-replication --bucket BUCKET_NAME --endpoint-
url=RADOSGW_ENDPOINT_URL
```

例

```
[root@host01 ~]# aws s3api get-bucket-replication --bucket testbucket --endpoint-
url=http://host01:80
{

"ReplicationConfiguration": {
    "Role": "arn:aws:iam::account-id:role/role-name",
    "Rules": [
        {
            "Status": "Enabled",
            "Priority": 1,
            "DeleteMarkerReplication": { "Status": "Disabled" },
            "Filter" : { "Prefix": ""},
            "Destination": {
                "Bucket": "testbucket"
            }
        }
    ]
 }

}
```

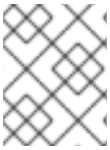
関連情報

- 詳細は、Red Hat Ceph Storage 開発者ガイドを参照してください。

8.8. バケットライフサイクル

ストレージ管理者では、バケットのライフサイクル設定を使用してオブジェクトを管理し、そのオブジェクトが有効期間中効果的に保存されるようにすることができます。たとえば、オブジェクトを、ユースケースに基づいて、コストの低いストレージクラス、アーカイブ、または削除にできます。

バケットオブジェクトのセットに定義されたルールを使用して、RADOS Gateway は S3 API オブジェクトの有効期限をサポートします。各ルールには、オブジェクトを選択する接頭辞と、オブジェクトが利用できなくなる日数が設定されます。



注記

radosgw-admin lc reshard コマンドは Red Hat Ceph Storage 3.3 で非推奨となり、Red Hat Ceph Storage 4 以降のリリースではサポートされません。

8.8.1. ライフサイクル管理ポリシーの作成

radosgw-admin コマンドを使用する代わりに、標準の S3 操作を使用してバケットのライフサイクルポリシー設定を管理できます。RADOS Gateway は、バケットに適用される Amazon S3 API ポリシー言語のサブセットのみをサポートします。ライフサイクル設定には、バケットオブジェクトのセットに定義される 1 つまたは複数のルールが含まれます。

前提条件

- 稼働中の Red Hat Storage クラスター
- Ceph Object Gateway のインストール
- Ceph Object Gateway ノードへのルートレベルのアクセス。
- S3 バケットが作成されている。
- ユーザーアクセスで作成された S3 ユーザー。
- **AWS CLI** パッケージがインストールされた Ceph Object Gateway クライアントへのアクセス。

手順

1. ライフサイクル設定用の JSON ファイルを作成します。

   例

   ```
   [user@client ~]$ vi lifecycle.json
   ```

2. ファイルに特定のライフサイクル設定ルールを追加します。

   例

   ```
   {
    "Rules": [
          {
        "Filter": {
         "Prefix": "images/"
        },
        "Status": "Enabled",
        "Expiration": {
         "Days": 1
        },
        "ID": "ImageExpiration"
       }
     ]
   }
   ```

   ライフサイクル設定の例では、1 日後に images ディレクトリーのオブジェクトの有効期限が切れます。

3. バケットにライフサイクル設定を設定します。

   構文

   ```
   aws --endpoint-url=RADOSGW_ENDPOINT_URL:PORT s3api put-bucket-lifecycle-configuration --bucket BUCKET_NAME --lifecycle-configuration file://PATH_TO_LIFECYCLE_CONFIGURATION_FILE/LIFECYCLE_CONFIGURATION_FILE.json
   ```

   例

```
[user@client ~]$ aws --endpoint-url=http://host01:80 s3api put-bucket-lifecycle-configuration
--bucket testbucket --lifecycle-configuration file://lifecycle.json
```

この例では、**lifecycle.json** ファイルが現在のディレクトリーに存在します。

検証

- バケットのライフサイクル設定を取得します。

  構文

  ```
  aws --endpoint-url=RADOSGW_ENDPOINT_URL:PORT s3api get-bucket-lifecycle-
  configuration --bucket BUCKET_NAME
  ```

  例

  ```
  [user@client ~]$ aws --endpoint-url=http://host01:80 s3api get-bucket-lifecycle-configuration
  --bucket testbucket
  {
   "Rules": [
         {
        "Expiration": {
         "Days": 1
        },
        "ID": "ImageExpiration",
        "Filter": {
         "Prefix": "images/"
        },
        "Status": "Enabled"
       }
      ]
  }
  ```

- オプション:Ceph Object Gateway ノードから、Cephadm シェルにログインし、バケットのライフサイクル設定を取得します。

  構文

  ```
  radosgw-admin lc get --bucket=BUCKET_NAME
  ```

  例

  ```
  [ceph: root@host01 /]# radosgw-admin lc get --bucket=testbucket
  {
   "prefix_map": {
    "images/": {
     "status": true,
     "dm_expiration": false,
     "expiration": 1,
     "noncur_expiration": 0,
     "mp_expiration": 0,
     "transitions": {},
     "noncur_transitions": {}
  ```

```
    }
   },
   "rule_map": [
        {
    "id": "ImageExpiration",
    "rule": {
     "id": "ImageExpiration",
     "prefix": "",
     "status": "Enabled",
     "expiration": {
      "days": "1",
      "date": ""
     },
     "mp_expiration": {
      "days": "",
      "date": ""
     },
     "filter": {
      "prefix": "images/",
      "obj_tags": {
       "tagset": {}
      }
     },
     "transitions": {},
     "noncur_transitions": {},
     "dm_expiration": false
    }
   }
    ]
}
```


関連情報

- 詳細は、**Red Hat Ceph Storage 開発者ガイド** の **S3 バケットライフサイクル** セクションを参照してください。
- **AWS CLI** を使用したライフサイクル設定の管理に関する詳細は、**Amazon Simple Storage Service** ドキュメントの **Setting lifecycle configuration on a bucket** セクションを参照してください。

8.8.2. ライフサイクル管理ポリシーの削除

s3api delete-bucket-lifecycle コマンドを使用すると、指定されたバケットのライフサイクル管理ポリシーを削除できます。

前提条件

- 稼働中の Red Hat Storage クラスター
- Ceph Object Gateway のインストール
- Ceph Object Gateway ノードへのルートレベルのアクセス。
- S3 バケットが作成されている。

- ユーザーアクセスで作成された S3 ユーザー。
- **AWS CLI** パッケージがインストールされた Ceph Object Gateway クライアントへのアクセス。

手順

- ライフサイクル設定を削除します。

  構文

  ```
  aws --endpoint-url=RADOSGW_ENDPOINT_URL:PORT s3api delete-bucket-lifecycle --bucket BUCKET_NAME
  ```

  例

  ```
  [user@client ~]$ aws --endpoint-url=http://host01:80 s3api delete-bucket-lifecycle --bucket testbucket
  ```

検証

- バケットのライフサイクル設定を取得します。

  構文

  ```
  aws --endpoint-url=RADOSGW_ENDPOINT_URL:PORT s3api get-bucket-lifecycle-configuration --bucket BUCKET_NAME
  ```

  例

  ```
  [user@client ~]# aws --endpoint-url=http://host01:80  s3api get-bucket-lifecycle-configuration --bucket testbucket
  ```

- オプション:Ceph Object Gateway ノードから、バケットのライフサイクル設定を取得します。

  構文

  ```
  radosgw-admin lc get --bucket=BUCKET_NAME
  ```

  例

  ```
  [ceph: root@host01 /]# radosgw-admin lc get --bucket=testbucket
  ```

  注記

  バケットのライフサイクルポリシーが存在しない場合、このコマンドは情報を返しません。

関連情報

- 詳細は、**Red Hat Ceph Storage 開発者ガイド** の **S3 バケットライフサイクル** セクションを参照してください。

8.8.3. ライフサイクル管理ポリシーの更新

s3cmd put-bucket-lifecycle-configuration コマンドを使用すると、ライフサイクル管理ポリシーを更新できます。

注記

put-bucket-lifecycle-configuration は、既存のバケットのライフサイクル設定を上書きします。現在のライフサイクルポリシー設定を保持する場合は、ライフサイクル設定ファイルに追加する必要があります。

前提条件

- 稼働中の Red Hat Storage クラスター
- Ceph Object Gateway のインストール
- Ceph Object Gateway ノードへのルートレベルのアクセス。
- S3 バケットが作成されている。
- ユーザーアクセスで作成された S3 ユーザー。
- **AWS CLI** パッケージがインストールされた Ceph Object Gateway クライアントへのアクセス。

手順

1. ライフサイクル設定用の JSON ファイルを作成します。

   例

   ```
   [user@client ~]$ vi lifecycle.json
   ```

2. ファイルに特定のライフサイクル設定ルールを追加します。

   例

   ```
   {
    "Rules": [
          {
        "Filter": {
         "Prefix": "images/"
        },
        "Status": "Enabled",
        "Expiration": {
         "Days": 1
        },
        "ID": "ImageExpiration"
       },
    {
     "Filter": {
   ```

```
    "Prefix": "docs/"
   },
   "Status": "Enabled",
   "Expiration": {
    "Days": 30
   },
   "ID": "DocsExpiration"
  }
 ]
}
```

3. バケットでライフサイクル設定を更新します。

構文

```
aws --endpoint-url=RADOSGW_ENDPOINT_URL:PORT s3api put-bucket-lifecycle-configuration --bucket BUCKET_NAME --lifecycle-configuration file://PATH_TO_LIFECYCLE_CONFIGURATION_FILE/LIFECYCLE_CONFIGURATION_FILE.json
```

例

```
[user@client ~]$ aws --endpoint-url=http://host01:80 s3api put-bucket-lifecycle-configuration --bucket testbucket --lifecycle-configuration file://lifecycle.json
```

検証

- バケットのライフサイクル設定を取得します。

構文

```
aws --endpointurl=RADOSGW_ENDPOINT_URL:PORT s3api get-bucket-lifecycle-configuration --bucket BUCKET_NAME
```

例

```
[user@client ~]$ aws -endpoint-url=http://host01:80 s3api get-bucket-lifecycle-configuration --bucket testbucket

{
    "Rules": [
        {
            "Expiration": {
                "Days": 30
            },
            "ID": "DocsExpiration",
            "Filter": {
                "Prefix": "docs/"
            },
            "Status": "Enabled"
        },
        {
            "Expiration": {
```

```
            "Days": 1
        },
        "ID": "ImageExpiration",
        "Filter": {
            "Prefix": "images/"
        },
        "Status": "Enabled"
    }
  ]
}
```

- オプション:Ceph Object Gateway ノードから、Cephadm シェルにログインし、バケットのライフサイクル設定を取得します。

  構文

  ```
  radosgw-admin lc get --bucket=BUCKET_NAME
  ```

  例

  ```
  [ceph: root@host01 /]# radosgw-admin lc get --bucket=testbucket
  {
   "prefix_map": {
         "docs/": {
    "status": true,
    "dm_expiration": false,
    "expiration": 1,
    "noncur_expiration": 0,
    "mp_expiration": 0,
    "transitions": {},
    "noncur_transitions": {}
   },
   "images/": {
    "status": true,
    "dm_expiration": false,
    "expiration": 1,
    "noncur_expiration": 0,
    "mp_expiration": 0,
    "transitions": {},
    "noncur_transitions": {}
   }
  },
  "rule_map": [
        {
         "id": "DocsExpiration",
      "rule": {
       "id": "DocsExpiration",
       "prefix": "",
       "status": "Enabled",
       "expiration": {
        "days": "30",
        "date": ""
       },
            "noncur_expiration": {
                "days": "",
  ```

```
            "date": ""
           },
      "mp_expiration": {
       "days": "",
       "date": ""
      },
      "filter": {
       "prefix": "docs/",
       "obj_tags": {
        "tagset": {}
       }
      },
      "transitions": {},
      "noncur_transitions": {},
      "dm_expiration": false
     }
    },
    {
   "id": "ImageExpiration",
   "rule": {
    "id": "ImageExpiration",
    "prefix": "",
    "status": "Enabled",
    "expiration": {
     "days": "1",
     "date": ""
    },
    "mp_expiration": {
     "days": "",
     "date": ""
    },
    "filter": {
     "prefix": "images/",
     "obj_tags": {
      "tagset": {}
     }
    },
    "transitions": {},
    "noncur_transitions": {},
    "dm_expiration": false
   }
  }
   ]
}
```


関連情報

- Amazon S3 バケットライフサイクルに関する詳細は、**Red Hat Ceph Storage 開発者ガイド**の Amazon S3 バケットライフサイクル を参照してください。

8.8.4. バケットライフサイクルのモニタリング

ライフサイクル処理を監視することや、**radosgw-admin lc list** および **radosgw-admin lc process** コマンドを使用してバケットのライフサイクルを手動で処理することができます。

前提条件

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- Ceph Object Gateway ノードへのルートレベルのアクセス。
- ライフサイクル設定ポリシーが適用される S3 バケットが作成されている。

手順

1. Cephadm シェルにログインします。

   例

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

2. バケットのライフサイクルの進捗をリスト表示します。

   例

   ```
   [ceph: root@host01 /]# radosgw-admin lc list

   [
     {
         “bucket”: “:testbucket:8b63d584-9ea1-4cf3-8443-a6a15beca943.54187.1”,
         “started”: “Thu, 01 Jan 1970 00:00:00 GMT”,
         “status” : “UNINITIAL”
     },
     {
         “bucket”: “:testbucket1:8b635499-9e41-4cf3-8443-a6a15345943.54187.2”,
         “started”: “Thu, 01 Jan 1970 00:00:00 GMT”,
         “status” : “UNINITIAL”
     }
   ]
   ```

   バケットのライフサイクル処理ステータスは、以下のいずれかになります。

   - UNINITIAL - プロセスはまだ実行されていません。
   - PROCESSING - プロセスは現在実行中です。
   - COMPLETE - プロセスが完了しました。

3. オプション: バケットのライフサイクルポリシーを手動で処理できます。

   a. 単一バケットのライフサイクルポリシーを処理します。

      構文

      ```
      radosgw-admin lc process --bucket=BUCKET_NAME
      ```

      例

      ```
      [ceph: root@host01 /]# radosgw-admin lc process --bucket=testbucket1
      ```

b. すべてのバケットのライフサイクルポリシーを即座に処理します。

例

```
[ceph: root@host01 /]# radosgw-admin lc process
```

検証

- バケットのライフサイクルポリシーをリスト表示します。

```
[ceph: root@host01 /]# radosgw-admin lc list
[
    {
        “bucket”: “:testbucket:8b63d584-9ea1-4cf3-8443-a6a15beca943.54187.1”,
        “started”: “Thu, 17 Mar 2022 21:48:50 GMT”,
        “status” : “COMPLETE”
    }
    {
        “bucket”: “:testbucket1:8b635499-9e41-4cf3-8443-a6a15345943.54187.2”,
        “started”: “Thu, 17 Mar 2022 20:38:50 GMT”,
        “status” : “COMPLETE”
    }
]
```

関連情報

- 詳細は、**Red Hat Ceph Storage 開発者ガイド** の **S3 バケットライフサイクル** セクションを参照してください。

8.8.5. ライフサイクルの有効期間の設定

rgw_lifecycle_work_time パラメーターを設定すると、毎日ライフサイクル管理プロセスが実行される時間を設定できます。デフォルトでは、ライフサイクルの処理は 1 日 1 回午前 0 時に行われます。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- Ceph Object Gateway のインストール
- Ceph Object Gateway ノードへのルートレベルのアクセス。

手順

1. Cephadm シェルにログインします。

   例

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

2. ライフサイクルの有効期限を設定します。

   構文

```
ceph config set client.rgw rgw_lifecycle_work_time %D:%D-%D:%D
```

%d:%d-%d:%d を **start_hour:start_minute-end_hour:end_minute** に置き換えます。

例

```
[ceph: root@host01 /]# ceph config set client.rgw rgw_lifecycle_work_time 06:00-08:00
```

検証

- ライフサイクル満了の作業時間を取得します。

  例

```
[ceph: root@host01 /]# ceph config get client.rgw rgw_lifecycle_work_time

06:00-08:00
```

関連情報

- 詳細は、**Red Hat Ceph Storage 開発者ガイド** の S3 バケットライフサイクル セクションを参照してください。

8.8.6. ストレージクラスター内での S3 バケットライフサイクルの移行

バケットライフサイクル設定を使用してオブジェクトを管理し、オブジェクトのライフタイム全体でオブジェクトを効果的に保存できます。オブジェクトライフサイクルの移行ルールを使用すると、オブジェクトのライフタイム全体でオブジェクトを管理し、効果的に保存できます。オブジェクトを、コストの低いストレージクラス、アーカイブ、または削除できます。

以下についてストレージクラスを作成できます。

- I/O 機密ワークロード用の SSD や NVMe などの高速メディア。
- アーカイブ用の SAS や SATA などの処理が遅いメディア。

ホットストレージクラスとコールドストレージクラスとの間で、データの移動をスケジュールできます。指定された時間後にこの移行をスケジュールして、オブジェクトの期限が切れ、永続的に削除されます。たとえば、オブジェクトを作成/からストレージクラスの作成後、またはストレージクラスを 1 年にアーカイブした後に、オブジェクトを 30 日間移行することができます。これは、移行ルールを使用して実行できます。このルールは、あるストレージクラスから別のストレージクラスに移行するオブジェクトに適用されます。ライフサイクル設定には、**<Rule>** 要素を使用した 1 つ以上のルールが含まれます。

関連情報

- バケットライフサイクル の詳細は、**Red Hat Ceph Storage 開発者ガイド**を参照してください。

8.8.7. あるストレージクラスから別のストレージクラスへのオブジェクトの移行

オブジェクトライフサイクルの移行ルールにより、オブジェクトをあるストレージクラスから別のストレージクラスに移行することができます。

Ceph Object Gateway ライフサイクル移行ポリシーを使用して、複製プール、イレージャーコーディングプール、複製プールからイレージャーコーディングプール、またはイレージャーコーディングプールから複製プールの間でデータを移行できます。

注記

マルチサイト設定では、ライフサイクル遷移ルールが最初のサイトに適用され、同じストレージクラスター内のあるデータプールから別のデータプールにオブジェクトを遷移する場合は、2 番目のサイトに **rgw** アプリケーションで作成され有効になっているそれぞれのデータプールがあれば、同じルールが 2 番目のサイトにも有効になります。

前提条件

- Ceph Object Gateway ソフトウェアのインストール。
- Ceph Object Gateway ノードへのルートレベルのアクセスがある。
- ユーザーアクセスで作成された S3 ユーザー。

手順

1. 新しいデータプールを作成します。

   構文

   ```
   ceph osd pool create POOL_NAME
   ```

   例

   ```
   [ceph: root@host01 /]# ceph osd pool create test.hot.data
   ```

2. 新規ストレージクラスを追加します。

   構文

   ```
   radosgw-admin zonegroup placement add  --rgw-zonegroup default --placement-id
   PLACEMENT_TARGET --storage-class STORAGE_CLASS
   ```

   例

   ```
   [ceph: root@host01 /]# radosgw-admin zonegroup placement add  --rgw-zonegroup default --
   placement-id default-placement --storage-class hot.test
   {
           "key": "default-placement",
           "val": {
               "name": "default-placement",
               "tags": [],
               "storage_classes": [
                   "STANDARD",
                   "hot.test"
               ]
           }
       }
   ```

3. 新規ストレージクラスのゾーン配置情報を提供します。

構文

```
radosgw-admin zone placement add --rgw-zone default --placement-id
PLACEMENT_TARGET --storage-class STORAGE_CLASS --data-pool DATA_POOL
```

例

```
[ceph: root@host01 /]# radosgw-admin zone placement add --rgw-zone default --placement-
id default-placement --storage-class hot.test --data-pool test.hot.data
{
        "key": "default-placement",
        "val": {
            "index_pool": "test_zone.rgw.buckets.index",
            "storage_classes": {
                "STANDARD": {
                    "data_pool": "test.hot.data"
                },
                "hot.test": {
                    "data_pool": "test.hot.data",
                }
            },
            "data_extra_pool": "",
            "index_type": 0
        }
```

注記

一度書き込みでコールドまたはアーカイブデータストレージプールを作成する際には、**compression_type** を設定することを検討してください。

4. データプールの **rgw** アプリケーションを有効にします。

構文

```
ceph osd pool application enable POOL_NAME rgw
```

例

```
[ceph: root@host01 /]# ceph osd pool application enable test.hot.data rgw
enabled application 'rgw' on pool 'test.hot.data'
```

5. すべての **rgw** デーモンを再起動します。

6. バケットを作成します。

例

```
[ceph: root@host01 /]# aws s3api create-bucket --bucket testbucket10 --create-bucket-
configuration LocationConstraint=default:default-placement --endpoint-url
http://10.0.0.80:8080
```

7. オブジェクトを追加します。

例

```
[ceph: root@host01 /]# aws --endpoint=http://10.0.0.80:8080 s3api put-object --bucket
testbucket10  --key compliance-upload --body /root/test2.txt
```

8. 2 番目のデータプールを作成します。

構文

```
ceph osd pool create POOL_NAME
```

例

```
[ceph: root@host01 /]# ceph osd pool create test.cold.data
```

9. 新規ストレージクラスを追加します。

構文

```
radosgw-admin zonegroup placement add  --rgw-zonegroup default --placement-id
PLACEMENT_TARGET --storage-class STORAGE_CLASS
```

例

```
[ceph: root@host01 /]# radosgw-admin zonegroup placement add  --rgw-zonegroup default --
placement-id default-placement --storage-class cold.test
{
        "key": "default-placement",
        "val": {
            "name": "default-placement",
            "tags": [],
            "storage_classes": [
                "STANDARD",
                "cold.test"
            ]
        }
    }
```

10. 新規ストレージクラスのゾーン配置情報を提供します。

構文

```
radosgw-admin zone placement add --rgw-zone default --placement-id
PLACEMENT_TARGET --storage-class STORAGE_CLASS --data-pool DATA_POOL
```

例

```
[ceph: root@host01 /]# radosgw-admin zone placement add --rgw-zone default --placement-
id default-placement --storage-class cold.test --data-pool test.cold.data
```

11. データプールの **rgw** アプリケーションを有効にします。

    構文

    ```
    ceph osd pool application enable POOL_NAME rgw
    ```

    例

    ```
    [ceph: root@host01 /]# ceph osd pool application enable test.cold.data rgw
    enabled application 'rgw' on pool 'test.cold.data'
    ```

12. すべての **rgw** デーモンを再起動します。

13. ゾーングループの設定を表示するには、次のコマンドを実行します。

    構文

    ```
    radosgw-admin zonegroup get
    {
        "id": "3019de59-ddde-4c5c-b532-7cdd29de09a1",
        "name": "default",
        "api_name": "default",
        "is_master": "true",
        "endpoints": [],
        "hostnames": [],
        "hostnames_s3website": [],
        "master_zone": "adacbe1b-02b4-41b8-b11d-0d505b442ed4",
        "zones": [
            {
                "id": "adacbe1b-02b4-41b8-b11d-0d505b442ed4",
                "name": "default",
                "endpoints": [],
                "log_meta": "false",
                "log_data": "false",
                "bucket_index_max_shards": 11,
                "read_only": "false",
                "tier_type": "",
                "sync_from_all": "true",
                "sync_from": [],
                "redirect_zone": ""
            }
        ],
        "placement_targets": [
            {
                "name": "default-placement",
                "tags": [],
                "storage_classes": [
                    "hot.test",
                    "cold.test",
                    "STANDARD"
                ]
            }
        ],
        "default_placement": "default-placement",
        "realm_id": "",
    ```

```
    "sync_policy": {
        "groups": []
    }
}
```

14. ゾーンの設定を表示するには、次のコマンドを実行します。

構文

```
radosgw-admin zone get
{
    "id": "adacbe1b-02b4-41b8-b11d-0d505b442ed4",
    "name": "default",
    "domain_root": "default.rgw.meta:root",
    "control_pool": "default.rgw.control",
    "gc_pool": "default.rgw.log:gc",
    "lc_pool": "default.rgw.log:lc",
    "log_pool": "default.rgw.log",
    "intent_log_pool": "default.rgw.log:intent",
    "usage_log_pool": "default.rgw.log:usage",
    "roles_pool": "default.rgw.meta:roles",
    "reshard_pool": "default.rgw.log:reshard",
    "user_keys_pool": "default.rgw.meta:users.keys",
    "user_email_pool": "default.rgw.meta:users.email",
    "user_swift_pool": "default.rgw.meta:users.swift",
    "user_uid_pool": "default.rgw.meta:users.uid",
    "otp_pool": "default.rgw.otp",
    "system_key": {
        "access_key": "",
        "secret_key": ""
    },
    "placement_pools": [
        {
            "key": "default-placement",
            "val": {
                "index_pool": "default.rgw.buckets.index",
                "storage_classes": {
                    "cold.test": {
                        "data_pool": "test.cold.data"
                    },
                    "hot.test": {
                        "data_pool": "test.hot.data"
                    },
                    "STANDARD": {
                        "data_pool": "default.rgw.buckets.data"
                    }
                },
                "data_extra_pool": "default.rgw.buckets.non-ec",
                "index_type": 0
            }
        }
    ],
    "realm_id": "",
    "notif_pool": "default.rgw.log:notif"
}
```

15. バケットを作成します。

例

```
[ceph: root@host01 /]# aws s3api create-bucket --bucket testbucket10 --create-bucket-configuration LocationConstraint=default:default-placement --endpoint-url http://10.0.0.80:8080
```

16. 遷移前のオブジェクトをリスト表示します。

例

```
[ceph: root@host01 /]# radosgw-admin bucket list --bucket testbucket10

        {
            "ETag": "\"211599863395c832a3dfcba92c6a3b90\"",
            "Size": 540,
            "StorageClass": "STANDARD",
            "Key": "obj1",
            "VersionId": "W95teRsXPSJI4YWJwwSG30KxSCzSgk-",
            "IsLatest": true,
            "LastModified": "2023-11-23T10:38:07.214Z",
            "Owner": {
                "DisplayName": "test-user",
                "ID": "test-user"
            }
        }
```

17. ライフサイクル設定用の JSON ファイルを作成します。

例

```
[ceph: root@host01 /]# vi lifecycle.json
```

18. ファイルに特定のライフサイクル設定ルールを追加します。

例

```
{
    "Rules": [
        {
            "Filter": {
                "Prefix": ""
            },
            "Status": "Enabled",
            "Transitions": [
                {
                    "Days": 5,
                    "StorageClass": "hot.test"
                },
 {
                    "Days": 20,
                    "StorageClass": "cold.test"
                }
```

```
            ],
            "Expiration": {
                "Days": 365
            },
            "ID": "double transition and expiration"
        }
    ]
}
```

ライフサイクル設定の例では、デフォルトの **STANDARD** ストレージクラスから 5 日後に **hot.test** ストレージクラスに移行し、20 日後に再び **cold.test** ストレージクラスに移行し、最後に **cold.test** ストレージクラスで 365 日後に期限切れとなるオブジェクトを示しています。

19. バケットにライフサイクル設定を設定します。

例

```
[ceph: root@host01 /]# aws s3api put-bucket-lifecycle-configuration --bucket testbucket10 --
lifecycle-configuration file://lifecycle.json
```

20. バケットでライフサイクル設定を取得します。

例

```
[ceph: root@host01 /]# aws s3api get-bucket-lifecycle-configuration --bucket testbucke10
{
    "Rules": [
        {
            "Expiration": {
                "Days": 365
            },
            "ID": "double transition and expiration",
            "Prefix": "",
            "Status": "Enabled",
            "Transitions": [
                {
                    "Days": 20,
                    "StorageClass": "cold.test"
                },
                {
                    "Days": 5,
                    "StorageClass": "hot.test"
                }
            ]
        }
    ]
}
```

21. オブジェクトが指定されたストレージクラスに移行されていることを確認します。

例

```
[ceph: root@host01 /]# radosgw-admin bucket list --bucket testbucket10

        {
```

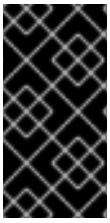
```
        "ETag": "\"211599863395c832a3dfcba92c6a3b90\"",
        "Size": 540,
        "StorageClass": "cold.test",
        "Key": "obj1",
        "VersionId": "W95teRsXPSJI4YWJwwSG30KxSCzSgk-",
        "IsLatest": true,
        "LastModified": "2023-11-23T10:38:07.214Z",
        "Owner": {
            "DisplayName": "test-user",
            "ID": "test-user"
        }
    }
```

関連情報

- バケットライフサイクル の詳細は、**Red Hat Ceph Storage 開発者ガイド**を参照してください。

8.8.8. S3 のオブジェクトロックの有効化

S3 オブジェクトロックメカニズムを使用すると、保持期間、正当な保持、バケット設定などのオブジェクトロックの概念を使用して、データ削除パーミッションを上書きするカスタムワークフローの一部として Write-Once-Read_Many (WORM) 機能を実装できます。



重要

オブジェクトバージョン (オブジェクト名ではありません) は、オブジェクトロックが正常に機能して **GOVERNANCE** または **COMPLIANCE** モードをサポートするための定義であり、必要な値です。オブジェクトの書き込み時にオブジェクトのバージョンを把握して、後で取得できるようにする必要があります。

前提条件

- Ceph Object Gateway がインストールされている実行中の Red Hat Ceph Storage クラスター。
- Ceph Object Gateway ノードへのルートレベルのアクセスがある。
- バージョンバケット作成権限を持つ S3 ユーザー。

手順

1. オブジェクトロックが有効なバケットを作成します。

   構文

   ```
   aws --endpoint=http://RGW_PORT:8080 s3api create-bucket --bucket BUCKET_NAME --object-lock-enabled-for-bucket
   ```

   例

   ```
   [root@rgw-2 ~]# aws --endpoint=http://rgw.ceph.com:8080 s3api create-bucket --bucket worm-bucket --object-lock-enabled-for-bucket
   ```

2. バケットの保持期間を設定します。

構文

```
aws --endpoint=http://RGW_PORT:8080 s3api put-object-lock-configuration --bucket BUCKET_NAME --object-lock-configuration '{ "ObjectLockEnabled": "Enabled", "Rule": { "DefaultRetention": { "Mode": "RETENTION_MODE", "Days": NUMBER_OF_DAYS }}}'
```

例

```
[root@rgw-2 ~]# aws --endpoint=http://rgw.ceph.com:8080 s3api put-object-lock-configuration --bucket worm-bucket --object-lock-configuration '{ "ObjectLockEnabled": "Enabled", "Rule": { "DefaultRetention": { "Mode": "COMPLIANCE", "Days": 10 }}}'
```

注記

S3 オブジェクトロックの RETENTION_MODE に GOVERNANCE モードまたは COMPLIANCE モードのいずれかを選択して、オブジェクトロックで保護されるオブジェクトバージョンに異なるレベルの保護を適用できます。

GOVERNANCE モードでは、特別なパーミッションがない限り、ユーザーはオブジェクトバージョンの上書きや削除、あるいはロック設定の変更を行うことはできません。

COMPLIANCE モードでは、AWS アカウントの root ユーザーを含め、保護されているオブジェクトバージョンを上書きまたは削除できるユーザーはいません。オブジェクトが COMPLIANCE モードでロックされると、RETENTION_MODE は変更できず、その保持期間を短くすることもできません。COMPLIANCE モードは、期間中にオブジェクトバージョンを上書きしたり、削除したりできないようにするのに役立ちます。

3. 保持期間が設定されたオブジェクトをバケットに配置します。

構文

```
aws --endpoint=http://RGW_PORT:8080 s3api put-object --bucket BUCKET_NAME --object-lock-mode RETENTION_MODE --object-lock-retain-until-date "DATE" --key compliance-upload --body TEST_FILE
```

例

```
[root@rgw-2 ~]# aws --endpoint=http://rgw.ceph.com:8080 s3api put-object --bucket worm-bucket --object-lock-mode COMPLIANCE --object-lock-retain-until-date "2022-05-31" --key compliance-upload --body test.dd
{
    "ETag": "\"d560ea5652951637ba9c594d8e6ea8c1\"",
    "VersionId": "Nhhk5kRS6Yp6dZXVWpZZdRcpSpBKToD"
}
```

4. 同じキーを使用して新規オブジェクトをアップロードします。

構文

```
aws --endpoint=http://RGW_PORT:8080 s3api put-object --bucket BUCKET_NAME --object-lock-mode RETENTION_MODE --object-lock-retain-until-date "DATE" --key compliance-upload --body PATH
```

例

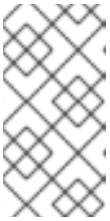
```
[root@rgw-2 ~]# aws --endpoint=http://rgw.ceph.com:8080 s3api put-object --bucket worm-bucket --object-lock-mode COMPLIANCE --object-lock-retain-until-date "2022-05-31" --key compliance-upload --body /etc/fstab
{
    "ETag": "\"d560ea5652951637ba9c594d8e6ea8c1\"",
    "VersionId": "Nhhk5kRS6Yp6dZXVWpZZdRcpSpBKToD"
}
```

コマンドラインオプション

- オブジェクトバージョンにオブジェクトロックの正当な保持を設定します。

  例

  ```
  [root@rgw-2 ~]# aws --endpoint=http://rgw.ceph.com:8080 s3api put-object-legal-hold --bucket worm-bucket --key compliance-upload --legal-hold Status=ON
  ```

  注記

  オブジェクトロックの正当な保持操作を使用すると、オブジェクトバージョンに正当な保持を配置できるため、オブジェクトバージョンの上書きや削除を防止することができます。正当な保持には保持期間が関連付けられていないため、削除されるまで有効になります。

- バケットからオブジェクトをリスト表示し、最新バージョンのオブジェクトのみを取得します。

  例

  ```
  [root@rgw-2 ~]# aws --endpoint=http://rgw.ceph.com:8080 s3api list-objects --bucket worm-bucket
  ```

- バケットからオブジェクトバージョンをリスト表示します。

  例

  ```
  [root@rgw-2 ~]# aws --endpoint=http://rgw.ceph.com:8080 s3api list-objects --bucket worm-bucket
  {
      "Versions": [
          {
              "ETag": "\"d560ea5652951637ba9c594d8e6ea8c1\"",
              "Size": 288,
              "StorageClass": "STANDARD",
              "Key": "hosts",
              "VersionId": "Nhhk5kRS6Yp6dZXVWpZZdRcpSpBKToD",
  ```

```
            "IsLatest": true,
            "LastModified": "2022-06-17T08:51:17.392000+00:00",
            "Owner": {
                "DisplayName": "Test User in Tenant test",
                "ID": "test$test.user"
            }
            }
        }
    ]
}
```

- version-id を使用してオブジェクトにアクセスします。

  例

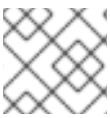
```
[root@rgw-2 ~]# aws --endpoint=http://rgw.ceph.com:8080 s3api get-object --bucket worm-bucket  --key compliance-upload --version-id 'IGOU.vdIs3SPduZglrB-RBaK.sfXpcd' download.1
{
    "AcceptRanges": "bytes",
    "LastModified": "2022-06-17T08:51:17+00:00",
    "ContentLength": 288,
    "ETag": "\"d560ea5652951637ba9c594d8e6ea8c1\"",
    "VersionId": "Nhhk5kRS6Yp6dZXVWpZZdRcpSpBKToD",
    "ContentType": "binary/octet-stream",
    "Metadata": {},
    "ObjectLockMode": "COMPLIANCE",
    "ObjectLockRetainUntilDate": "2023-06-17T08:51:17+00:00"
}
```

8.9. 使用方法

Ceph Object Gateway は、各ユーザーの使用状況をログに記録します。ユーザーの使用状況を日付の範囲内でも追跡できます。

オプションには以下が含まれます。

- **開始日: --start-date** オプションを使用すると、特定の開始日から使用統計をフィルタリングできます (**形式: yyyy-mm-dd[HH:MM:SS]**)。
- **終了日: --end-date** オプションを使用すると、特定の日付までの使用をフィルタリングできます (**形式: yyyy-mm-dd[HH:MM:SS]**)。
- **ログエントリー: --show-log-entries** オプションを使用すると、ログエントリーを使用統計に含めるかどうかを指定できます (オプション: **true** | **false**)。



注記

分と秒で時間を指定できますが、1時間分解能で保存されます。

8.9.1. 使用方法の表示

使用状況の統計を表示するには、**usage show** を指定します。特定のユーザーの使用状況を表示するには、ユーザー ID を指定する必要があります。開始日、終了日、およびログエントリーを表示するかどうかを指定することもできます。

例

```
[ceph: root@host01 /]# radosgw-admin usage show \
            --uid=johndoe --start-date=2022-06-01 \
            --end-date=2022-07-01
```

ユーザー ID を省略することで、すべてのユーザーの使用状況情報の概要も表示できます。

例

```
[ceph: root@host01 /]# radosgw-admin usage show --show-log-entries=false
```

8.9.2. トリムの使用方法

頻繁に使用すると、使用状況のログがストレージスペースを占有し始める可能性があります。すべてのユーザーおよび特定ユーザーの使用状況ログをトリミングできます。トリム操作の日付範囲を指定することもできます。

例

```
[ceph: root@host01 /]# radosgw-admin usage trim --start-date=2022-06-01 \
              --end-date=2022-07-31

[ceph: root@host01 /]# radosgw-admin usage trim --uid=johndoe
[ceph: root@host01 /]# radosgw-admin usage trim --uid=johndoe --end-date=2021-04-31
```

8.10. CEPH OBJECT GATEWAY データレイアウト

RADOS は拡張属性 (**xattrs**) とオブジェクトマップ (OMAP) を持つプールとオブジェクトしか認識しませんが、概念的には、Ceph Object Gateway はそのデータを 3 つの異なる種類に編成します。

- metadata
- バケットインデックス
- data

メタデータ

メタデータには 3 つのセクションがあります。

- **user**: ユーザー情報を保持します。
- **bucket**: バケット名とバケットインスタンス ID の間のマッピングを保持します。
- **bucket.instance**: バケットインスタンス情報を保持します。

以下のコマンドを使用して、メタデータエントリーを表示できます。

構文

```
radosgw-admin metadata get bucket:BUCKET_NAME
radosgw-admin metadata get bucket.instance:BUCKET:BUCKET_ID
radosgw-admin metadata get user:USER
radosgw-admin metadata set user:USER
```

例

```
[ceph: root@host01 /]# radosgw-admin metadata list
[ceph: root@host01 /]# radosgw-admin metadata list bucket
[ceph: root@host01 /]# radosgw-admin metadata list bucket.instance
[ceph: root@host01 /]# radosgw-admin metadata list user
```

すべてのメタデータエントリーは単一の RADOS オブジェクトに保存されます。

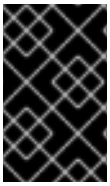
注記

Ceph Object Gateway オブジェクトは、いくつかの RADOS オブジェクトで設定される場合があります。最初のオブジェクトは、マニフェスト、アクセス制御リスト (ACL)、コンテンツタイプ、ETag、およびユーザー定義のメタデータなどのメタデータなどの Head です。メタデータは **xattrs** に保存されます。また、効率性とアトミック性を確保するために、最大 512 KB のオブジェクトデータが含まれる場合もあります。マニフェストには、各オブジェクトが RADOS オブジェクトでどのように配置されているかが記述されています。

バケットインデックス

これは別の種類のメタデータであり、個別に保持されます。バケットインデックスは RADOS オブジェクトのキーと値マップを保持します。デフォルトでは、バケットごとに 1 つの RADOS オブジェクトですが、複数の RADOS オブジェクトにわたってマップをシャードすることができます。

マップ自体は、各 RADOS オブジェクトに関連付けられた OMAP に保持されます。各 OMAP のキーはオブジェクトの名前であり、値にはそのオブジェクトの基本的なメタデータ (バケットを一覧表示するときに表示されるメタデータ) が保持されます。各 OMAP はヘッダーを保持し、オブジェクトの数、合計サイズなど、そのヘッダーにバケットアカウンティングメタデータを保持します。



重要

radosgw-admin ツールを使用する場合は、ツールと Ceph クラスターが同じバージョンであることを確認してください。一致しないバージョンの使用は**サポートされていません**。

注記

OMAP は、拡張属性が POSIX ファイルに関連付けられているのと同様に、オブジェクトに関連付けられたキー値ストアです。オブジェクトの OMAP はオブジェクトのストレージに物理的に配置されていませんが、その正確な実装は不可視であり、Ceph Object Gateway には重要ではありません。

Data

オブジェクトデータは、各 Ceph Object Gateway オブジェクトの 1 つ以上の RADOS オブジェクトに保持されます。

8.10.1. オブジェクトルックアップパス

オブジェクトにアクセスする場合、REST API は 3 つのパラメーターを持つ Ceph Object Gateway に送られます。

- S3 のアクセスキーまたは Swift のアカウント名を持つアカウント情報
- バケットまたはコンテナー名
- オブジェクト名またはキー

現在、Ceph Object Gateway はアカウント情報のみを使用してユーザー ID とアクセス制御を検索します。バケット名とオブジェクトキーのみを使用してプールのオブジェクトに対応します。

アカウント情報

Ceph Object Gateway のユーザー ID は文字列であり、通常はユーザー認証情報からの実際のユーザー名であり、ハッシュまたはマップされた識別子ではありません。

ユーザーのデータにアクセスする場合、ユーザーレコードは namespace が **users.uid** の **default.rgw.meta** プールのオブジェクト **USER_ID** から読み込まれます。.バケット名 **default.rgw.meta** プールで **root** 名前空間で表現されます。バケット ID として機能するマーカーを取得するために、バケットレコードがロードされます。

オブジェクト名

オブジェクトは **default.rgw.buckets.data** プールにあります。オブジェクト名は **MARKER_KEY** です。たとえば、**default.7593.4_image.png** の場合、マーカーは **default.7593.4** で、キーは **image.png** です。これらの連結された名前は解析されず、RADOS のみに渡されます。したがって、セパレーターの選択は重要ではなく、あいまいさを生じさせることはありません。同じ理由で、キーなどのオブジェクト名ではスラッシュを使用できます。

8.10.1.1. 複数のデータプール

複数のデータプールを作成して、異なるユーザーのバケットがデフォルトで異なる RADOS プールに作成されるようにすることで、必要なスケーリングを実現できます。これらのプールのレイアウトと命名は、**ポリシー** 設定によって制御されます。

8.10.2. バケットおよびオブジェクトの一覧

特定のユーザーに属するバケットは、namespace が **users.uid** の **default.rgw.meta** プール内の **foo.buckets** など、**USER_ID.buckets** という名前のオブジェクトの OMAP にリストされます。これらのオブジェクトには、バケットの一覧表示時、バケットの内容の更新時、およびクォータなどのバケット統計の更新および取得時にアクセスされます。これらのリストは、**.rgw** プール内のバケットと一貫性が保たれています。

注記

これらの OMAP エントリーの値は、ユーザーに表示されるエンコードされたクラス **cls_user_bucket_entry** およびネストされたクラス **cls_user_bucket** を参照してください。

指定のバケットに属するオブジェクトはバケットインデックスに一覧表示されます。インデックスオブジェクトのデフォルトの命名は、**default.rgw.buckets.index** プールの **.dir.MARKER** です。

関連情報

- 詳細は、Red Hat Ceph Storage Object Gateway ガイドの バケットインデックスのリシャーディングの設定 セクションを参照してください。

8.10.3. Object Gateway データレイアウトパラメーター

これは、Ceph Object Gateway のデータレイアウトパラメーターの一覧です。

既知のプール:

.rgw.root

オブジェクトごとに 1 つの、指定されていない地域、ゾーン、およびグローバル情報レコード。

ZONE.rgw.control

notify.**N**

ZONE.rgw.meta

さまざまな種類のメタデータを持つ複数の namespace

Namespace: root

BUCKET .bucket.meta.**BUCKET**:**MARKER** # see put_bucket_instance_info()
テナントはバケットを明確にするために使用されますが、バケットインスタンスには使用されません。

例

```
.bucket.meta.prodtx:test%25star:default.84099.6
.bucket.meta.testcont:default.4126.1
.bucket.meta.prodtx:testcont:default.84099.4
prodtx/testcont
prodtx/test%25star
testcont
```

namespace: users.uid

USER オブジェクト内のユーザーごとの情報 (RGWUserInfo) と、**USER.buckets** オブジェクトの omap 内のバケットのユーザーごとのリストの両方が含まれます。空でない場合には、**USER** にそのテナントが含まれる場合があります。

例

```
prodtx$prodt
test2.buckets
prodtx$prodt.buckets
test2
```

namespace: users.email

Unimportant

namespace: users.keys

47UA98JSTJZ9YAN3OS3O
これにより、Ceph Object Gateway は認証中にアクセスキーでユーザーを検索できます。

namespace: users.swift

test:tester

ZONE.rgw.buckets.index

オブジェクトの名前は **.dir.MARKER** で、それぞれにバケットインデックスが含まれます。インデックスがシャード化されている場合、各シャードはマーカーの後にシャードインデックスを追加します。

ZONE.rgw.buckets.data

default.7593.4__shadow_.488urDFerTYXavx4yAd-Op8mxehnvTI_1 MARKER_KEY
マーカーの例は、**default.16004.1** または **default.7593.4** です。現在の形式は **ZONE です。INSTANCE_ID。BUCKET_ID** ですが、一度生成されたマーカーは再度解析されないため、今後その形式が自由に変更される可能性があります。

関連情報

- 詳細は、**Red Hat Ceph Storage Object Gateway ガイド** の Ceph Object Gateway データレイアウト を参照してください。

8.11. データ取り込みのレート制限

ストレージ管理者は、Ceph Object Gateway 設定を使用して Red Hat Ceph Storage クラスターにオブジェクトを保存するときに、操作と帯域幅に基づいてユーザーとバケットにレート制限を設定できます。

8.11.1. ストレージクラスターでのレート制限の目的

Ceph Object Gateway 設定でユーザーとバケットにレート制限を設定できます。レート制限には、読み取り操作の最大数、1 分あたりの書き込み操作、およびユーザーごとまたはバケットごとに 1 分あたりに書き込みまたは読み取りできるバイト数が含まれます。

REST で GET または HEAD メソッドを使用するリクエストは読み取りリクエストであり、それ以外は書き込みリクエストです。

Ceph Object Gateway は、ユーザーとバケットのリクエストを別々に追跡し、他のゲートウェイとは共有しません。つまり、設定された目的の制限をアクティブな Object Gateway の数で割る必要があります。

たとえば、ユーザー A が 1 分あたり 10 操作に制限され、クラスター内に 2 つの Ceph Object Gateway がある場合、ユーザー A に対する制限は 5 である必要があります。つまり、2 つの Ceph Object Gateway に対して 1 分あたり 10 操作です。リクエストが Ceph Object Gateway 間でバランスが取れていない場合、レート制限が十分に活用されていない可能性があります。たとえば、ops の制限が 5 で、Ceph Object Gateway が 2 つあるが、ロードバランサーがこれらの Ceph Object Gateway の 1 つだけに負荷を送信する場合、この制限は Ceph Object Gateway ごとに適用されるため、有効な制限は 5 ops になります。

バケットの制限に達してもユーザーの制限に達しない場合、またはその逆の場合、リクエストもキャンセルされます。

帯域幅のカウントは、要求が受け入れられた後に行われます。その結果、リクエストの途中でバケットまたはユーザーが帯域幅の制限に達した場合でも、このリクエストは続行されます。

Ceph Object Gateway は、設定された値よりも多くの使用済みバイトの「負債」を保持し、「負債」が支払われるまで、このユーザーまたはバケットがそれ以上リクエストを送信できないようにします。「債務」の最大サイズは、1 分あたりの最大読み取り/書き込みバイト数の 2 倍です。ユーザー A に 1 分あたり 1 バイトの読み取り制限があり、このユーザーが 1 GB オブジェクトを取得しようとすると、ユーザーはそれを実行できます。

ユーザー A がこの 1 GB 操作を完了すると、Ceph Object Gateway は、ユーザー A が GET 要求を再度送信できるようになるまで、最大 2 分間ユーザー要求をブロックします。

レートを制限するためのさまざまなオプション:

- バケット: **--bucket** オプションを使用すると、バケットのレート制限を指定できます。
- ユーザー: **--uid** オプションを使用すると、ユーザーのレート制限を指定できます。
- 最大読み取り操作: **--max-read-ops** 設定を使用すると、Ceph Object Gateway ごとに 1 分あたりの読み取り操作の最大数を指定できます。値 **0** はこの設定を無効にし、無制限のアクセスを意味します。
- 最大読み取りバイト数: **--max-read-bytes** 設定を使用すると、Ceph Object Gateway ごとに 1 分あたりの最大読み取りバイト数を指定できます。値 **0** はこの設定を無効にし、無制限のアクセスを意味します。
- 最大書き込み操作: **--max-write-ops** 設定を使用すると、Ceph Object Gateway ごとに 1 分あたりの書き込み操作の最大数を指定できます。値 **0** はこの設定を無効にし、無制限のアクセスを意味します。
- 最大書き込みバイト数: **--max-write-bytes** 設定を使用すると、Ceph Object Gateway ごとに 1 分あたりの最大書き込みバイト数を指定できます。値 **0** はこの設定を無効にし、無制限のアクセスを意味します。
- レート制限スコープ: **--rate-limit-scope** オプションは、レート制限のスコープを設定します。オプションは、**bucket**、**user**、および **anonymous** です。バケットレート制限はバケットに適用され、ユーザーレート制限はユーザーに適用され、匿名は認証されていないユーザーに適用されます。匿名スコープは、グローバルレート制限でのみ使用できます。

8.11.2. ユーザーレート制限を有効にする

Ceph Object Gateway 設定でユーザーにレート制限を設定できます。ユーザーのレート制限には、読み取り操作の最大数、1 分あたりの書き込み操作、およびユーザーごとに 1 分あたりに書き込みまたは読み取りできるバイト数が含まれます。

ratelimit-scope を **user** に設定して **radosgw-admin ratelimit set** コマンドを使用することにより、レート制限の値を設定した後で、ユーザーのレート制限を有効にすることができます。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- Ceph Object Gateway がインストールされている。

手順

1. ユーザーのレート制限を設定します。

   構文

   ```
   radosgw-admin ratelimit set --ratelimit-scope=user --uid=USER_ID [--max-read-ops=NUMBER_OF_OPERATIONS] [--max-read-bytes=NUMBER_OF_BYTES] [--max-write-ops=NUMBER_OF_OPERATIONS] [--max-write-bytes=NUMBER_OF_BYTES]
   ```

例

```
[ceph: root@host01 /]# radosgw-admin ratelimit set --ratelimit-scope=user --uid=testing --
max-read-ops=1024 --max-write-bytes=10240
```

NUMBER_OF_OPERATIONS または NUMBER_OF_BYTES の値 **0** は、特定のレート制限属性チェックが無効であることを意味します。

2. ユーザーのレート制限を取得します。

構文

```
radosgw-admin ratelimit get --ratelimit-scope=user --uid=USER_ID
```

例

```
[ceph: root@host01 /]# radosgw-admin ratelimit get --ratelimit-scope=user --uid=testing

{
    "user_ratelimit": {
        "max_read_ops": 1024,
        "max_write_ops": 0,
        "max_read_bytes": 0,
        "max_write_bytes": 10240,
        "enabled": false
    }
}
```

3. ユーザーレート制限を有効にします。

構文

```
radosgw-admin ratelimit enable --ratelimit-scope=user --uid=USER_ID
```

例

```
[ceph: root@host01 /]# radosgw-admin ratelimit enable --ratelimit-scope=user --uid=testing

{
    "user_ratelimit": {
        "max_read_ops": 1024,
        "max_write_ops": 0,
        "max_read_bytes": 0,
        "max_write_bytes": 10240,
        "enabled": true
    }
}
```

4. オプション: ユーザーのレート制限を無効にします。

構文

```
radosgw-admin ratelimit disable --ratelimit-scope=user --uid=USER_ID
```

例

```
[ceph: root@host01 /]# radosgw-admin ratelimit disable --ratelimit-scope=user --uid=testing
```

8.11.3. バケットレート制限の有効化

Ceph Object Gateway 設定でバケットにレート制限を設定できます。バケットのレート制限には、読み取り操作の最大数、1分あたりの書き込み操作、およびユーザーごとに1分あたりに書き込みまたは読み取りできるバイト数が含まれます。

ratelimit-scope を **bucket** に設定して **radosgw-admin ratelimit set** コマンドを使用することにより、レート制限の値を設定した後で、バケットのレート制限を有効にすることができます。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- Ceph Object Gateway がインストールされている。

手順

1. バケットのレート制限を設定します。

   構文

   ```
   radosgw-admin ratelimit set --ratelimit-scope=bucket --bucket= BUCKET_NAME [--max-read-ops=NUMBER_OF_OPERATIONS] [--max-read-bytes=NUMBER_OF_BYTES] [--max-write-ops=NUMBER_OF_OPERATIONS] [--max-write-bytes=NUMBER_OF_BYTES]
   ```

   例

   ```
   [ceph: root@host01 /]# radosgw-admin ratelimit set --ratelimit-scope=bucket --bucket=mybucket --max-read-ops=1024 --max-write-bytes=10240
   ```

   NUMBER_OF_OPERATIONS または NUMBER_OF_BYTES の値 **0** は、特定のレート制限属性チェックが無効であることを意味します。

2. バケットレート制限を取得します。

   構文

   ```
   radosgw-admin ratelimit get --ratelimit-scope=bucket --bucket=BUCKET_NAME
   ```

   例

   ```
   [ceph: root@host01 /]# radosgw-admin ratelimit get --ratelimit-scope=bucket --bucket=mybucket

   {
       "bucket_ratelimit": {
           "max_read_ops": 1024,
           "max_write_ops": 0,
           "max_read_bytes": 0,
   ```

```
        "max_write_bytes": 10240,
        "enabled": false
    }
}
```

3. バケットレート制限を有効にします。

構文

```
radosgw-admin ratelimit enable --ratelimit-scope=bucket --bucket=BUCKET_NAME
```

例

```
[ceph: root@host01 /]# radosgw-admin ratelimit enable --ratelimit-scope=bucket --
bucket=mybucket

{
    "bucket_ratelimit": {
        "max_read_ops": 1024,
        "max_write_ops": 0,
        "max_read_bytes": 0,
        "max_write_bytes": 10240,
        "enabled": true
    }
}
```

4. オプション: バケットのレート制限を無効にします。

構文

```
radosgw-admin ratelimit disable --ratelimit-scope=bucket --bucket=BUCKET_NAME
```

例

```
[ceph: root@host01 /]# radosgw-admin ratelimit disable --ratelimit-scope=bucket --
bucket=mybucket
```

8.11.4. グローバルレート制限の設定

期間設定でグローバルレート制限設定を読み書きできます。グローバルレート制限パラメーターを使用して **global ratelimit** 設定を操作することにより、ユーザーまたはバケットのレート制限設定を上書きできます。これは、**ratelimit set**、**ratelimit enable**、および **ratelimit disable** コマンドに相当します。

注記

レルムと期間が存在するマルチサイト設定では、グローバルレート制限への変更は、**period update --commit** コマンドを使用してコミットする必要があります。期間が表示されていない場合、変更を有効にするには、Ceph Object Gateway を再起動する必要があります。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- Ceph Object Gateway がインストールされている。

手順

1. グローバルレート制限設定を表示します。

   構文

   ```
   radosgw-admin global ratelimit get
   ```

   例

   ```
   [ceph: root@host01 /]# radosgw-admin global ratelimit get

   {
       "bucket_ratelimit": {
           "max_read_ops": 1024,
           "max_write_ops": 0,
           "max_read_bytes": 0,
           "max_write_bytes": 0,
           "enabled": false
       },
       "user_ratelimit": {
           "max_read_ops": 0,
           "max_write_ops": 0,
           "max_read_bytes": 0,
           "max_write_bytes": 0,
           "enabled": false
       },
       "anonymous_ratelimit": {
           "max_read_ops": 0,
           "max_write_ops": 0,
           "max_read_bytes": 0,
           "max_write_bytes": 0,
           "enabled": false
       }
   }
   ```

2. バケットのレート制限スコープを設定して有効にします。

   a. バケットのグローバルレート制限を設定します。

      構文

      ```
      radosgw-admin global ratelimit set --ratelimit-scope=bucket [--max-read-ops=NUMBER_OF_OPERATIONS] [--max-read-bytes=NUMBER_OF_BYTES] [--max-write-ops=NUMBER_OF_OPERATIONS] [--max-write-bytes=NUMBER_OF_BYTES]
      ```

      例

```
[ceph: root@host01 /]# radosgw-admin global ratelimit set --ratelimit-scope bucket --max-read-ops=1024
```

b. バケットレート制限を有効にします。

構文

```
radosgw-admin global ratelimit enable --ratelimit-scope=bucket
```

例

```
[ceph: root@host01 /]# radosgw-admin global ratelimit enable --ratelimit-scope bucket
```

3. 認証済みユーザーのレート制限スコープを設定して有効にします。

a. ユーザーのグローバルレート制限を設定します。

構文

```
radosgw-admin global ratelimit set --ratelimit-scope=user [--max-read-ops=NUMBER_OF_OPERATIONS] [--max-read-bytes=NUMBER_OF_BYTES] [--max-write-ops=NUMBER_OF_OPERATIONS] [--max-write-bytes=NUMBER_OF_BYTES]
```

例

```
[ceph: root@host01 /]# radosgw-admin global ratelimit set --ratelimit-scope=user --max-read-ops=1024
```

b. ユーザーレート制限を有効にします。

構文

```
radosgw-admin global ratelimit enable --ratelimit-scope=user
```

例

```
[ceph: root@host01 /]# radosgw-admin global ratelimit enable --ratelimit-scope=user
```

4. 認証されていないユーザーのレート制限スコープを設定して有効にします。

a. 認証されていないユーザーのグローバルレート制限を設定します。

構文

```
radosgw-admin global ratelimit set --ratelimit-scope=anonymous [--max-read-ops=NUMBER_OF_OPERATIONS] [--max-read-bytes=NUMBER_OF_BYTES] [--max-write-ops=NUMBER_OF_OPERATIONS] [--max-write-bytes=NUMBER_OF_BYTES]
```

例

```
[ceph: root@host01 /]# radosgw-admin global ratelimit set --ratelimit-scope=anonymous --max-read-ops=1024
```

b. ユーザーレート制限を有効にします。

構文

```
radosgw-admin global ratelimit enable --ratelimit-scope=anonymous
```

例

```
[ceph: root@host01 /]# radosgw-admin global ratelimit enable --ratelimit-scope=anonymous
```

8.12. CEPH OBJECT GATEWAY のガベージコレクションの最適化

新規データオブジェクトがストレージクラスターに書き込まれると、Ceph Object Gateway はこれらの新規オブジェクトにすぐにストレージを割り当てます。ストレージクラスターのデータオブジェクトを削除または上書きした後に、Ceph Object Gateway はバケットインデックスからこれらのオブジェクトを削除します。その後しばらくして、Ceph Object Gateway は、ストレージクラスターにオブジェクトを格納するために使用されたスペースをパージします。ストレージクラスターから削除されたオブジェクトデータをパージするプロセスは、ガベージコレクションまたは GC として知られています。

通常、ガベージコレクションの操作はバックグラウンドで実行されます。これらの操作は、継続的に実行するように設定することも、アクティビティーが少なくワークロードが軽い期間でのみ実行するように設定することもできます。デフォルトでは、Ceph Object Gateway は GC 操作を継続的に実行します。GC 操作は Ceph Object Gateway 操作の通常の部分であるため、ガベージコレクションの対象となる削除済みオブジェクトはほとんどの場合存在します。

8.12.1. ガベージコレクションキューの表示

削除および上書きされたオブジェクトをストレージクラスターからパージする前に、**radosgw-admin** を使用して、ガベージコレクションを待機しているオブジェクトを表示します。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- Ceph Object Gateway へのルートレベルのアクセス。

手順

- ガベージコレクションを待っているオブジェクトのキューを表示するには、以下を実行します。

  例

  ```
  [ceph: root@host01 /]# radosgw-admin gc list
  ```

注記

有効期限が切れていないエントリーを含む、キュー内のすべてのエントリーを表示するには、**--include-all** オプションを使用します。

8.12.2. ガベージコレクション設定の調整

Ceph Object Gateway は、新規および上書きされたオブジェクトのストレージをすぐに割り当てます。また、マルチパートアップロードの一部も一部のストレージを使用します。

Ceph Object Gateway は、バケットインデックスからオブジェクトを削除した後に、削除されたオブジェクトに使用されるストレージ領域をパージします。同様に、Ceph Object Gateway は、マルチパートアップロードの完了後にマルチパートアップロードに関連付けられたデータを削除します。または、アップロードが非アクティブであるか、設定可能な期間完了に失敗した場合には、これを削除します。Red Hat Ceph Storage クラスターから削除されたオブジェクトデータをパージするプロセスは、ガベージコレクション (GC) と呼ばれます。

ガベージコレクション待ちのオブジェクトを表示するには、以下のコマンドを実行します。

```
radosgw-admin gc list
```

ガベージコレクションは、ストレージ管理者が Ceph Object Gateway をどのように設定するかに応じて、負荷の低いタイミングで継続的に実行するバックグラウンドアクティビティーです。デフォルトでは、Ceph Object Gateway はガベージコレクション操作を継続的に実行します。ガベッジコレクションの操作は Ceph Object Gateway の通常の機能であるため、特にオブジェクト削除操作では、ガベージコレクションの対象となるオブジェクトはほとんどの場合存在します。

一部のワークロードは、一時的または永続的にガベージコレクションのアクティビティーのレートをアウトプレートできます。これは、多くのオブジェクトが短期間保存され、その後に削除される、削除の多いワークロードが特に当てはまります。これらのタイプのワークロードでは、ストレージ管理者は、以下の設定パラメーターで、他の操作に関連したガベージコレクション操作の優先度を増やすことができます。

- **rgw_gc_obj_min_wait** 設定オプションは、削除されたオブジェクトのデータをパージする前に待機する最小時間 (秒単位) です。デフォルト値は 2 時間 (7200 秒) です。クライアントはオブジェクトを読み取る可能性があるため、オブジェクトはすぐにパージされません。負荷が大きい場合、この設定は過剰なストレージを消費するか、パージする多数のオブジェクトが存在する可能性があります。Red Hat は、この値を 30 分以下の 1800 秒に設定しないことを推奨します。

- **rgw_gc_processor_period** 設定オプションは、ガベージコレクションサイクルのランタイムです。つまり、ガベージコレクションスレッドが連続して実行されるまでの時間です。ガベージコレクションがこの期間よりも長く実行される場合、Ceph Object Gateway はガベージコレクションサイクルを再度実行する前に待機しません。

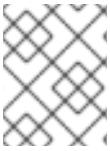
- **rgw_gc_max_concurrent_io** 設定オプションは、削除されたデータをパージする際にゲートウェイガベッジコレクションスレッドが使用する同時 IO 操作の最大数を指定します。負荷が大きい場合には、この設定を多数の同時 IO 操作に増やすことを検討してください。

- **rgw_gc_max_trim_chunk** 設定オプションは、単一の操作でガベージコレクターログから削除する鍵の最大数を指定します。大きな操作を削除する場合、それぞれのガベージコレクション操作中により多くのオブジェクトがパージされるようにキーの最大数を増やすことを検討してください。

Red Hat Ceph Storage 4.1 以降、ガベッジコレクションログからインデックスオブジェクトの OMAP をオフロードすると、ストレージクラスターのガベージコレクションアクティビティーのパフォーマンス

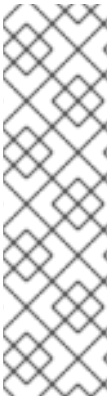
への影響が低下します。Ceph Object Gateway に新たな設定パラメーターが追加され、以下のようにガベージコレクションキューを調整します。

- The **rgw_gc_max_deferred_entries_size** 設定オプションは、ガベージコレクションのキューに遅延エントリーの最大サイズを設定します。
- **rgw_gc_max_queue_size** 設定オプションは、ガベッジコレクションに使用する最大キューサイズを設定します。この値は、**osd_max_object_size** から **rgw_gc_max_deferred_entries_size** を引いた値から 1KB を引いた値よりも大きくすることはできません。
- **rgw_gc_max_deferred** 設定オプションは、ガベージコレクションキューに保存された遅延エントリーの最大数を設定します。



注記

これらのガベージコレクション設定パラメーターは、Red Hat Ceph Storage 7 以降用です。



注記

テストでは、50% の削除操作と 50% の書き込み操作など、均等にバランスの取れた削除/書き込みワークロードを使用すると、ストレージクラスターは 11 時間で完全にいっぱいになります。これは、Ceph Object Gateway のガベージコレクションの削除操作によるペースの保持に失敗するためです。この場合、クラスターのステータスは **HEALTH_ERR** 状態に切り替わります。並列ガベッジコレクションの調整可能パラメーターの設定項目の設定により、テストするストレージクラスターセットが大幅に遅延し、多くのワークロードに役立ちます。典型的な実際のストレージクラスターワークロードは、主にガベージコレクションによりストレージクラスターがいっぱいになる可能性は想定されていません。

8.12.3. 削除が多いワークロードのガベージコレクションの調整

一部のワークロードは、一時的または永続的にガベージコレクションのアクティビティーの回数を上回る場合があります。これは、多くのオブジェクトが短期間保存されてから削除される、削除の多いワークロードに特に当てはまります。これらのタイプのワークロードでは、他の操作と比較してガベージコレクション操作の優先度を上げることを検討してください。Ceph Object Gateway ガベージコレクションに関するその他の質問については、Red Hat サポートにお問い合わせください。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- ストレージクラスター内のすべてのノードへの root レベルのアクセス。

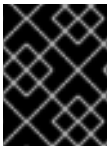
手順

1. **rgw_gc_max_concurrent_io** の値を **20** に設定し、**rgw_gc_max_trim_chunk** の値を **64** に設定します。

   例

```
[ceph: root@host01 /]# ceph config set client.rgw rgw_gc_max_concurrent_io 20
[ceph: root@host01 /]# ceph config set client.rgw rgw_gc_max_trim_chunk 64
```

2. Ceph Object Gateway を再起動して、変更した設定が有効になるようにします。
3. GC アクティビティー中にストレージクラスターを監視して、値の増加がパフォーマンスに悪影響を与えないことを確認します。



重要

実行中のクラスターの **rgw_gc_max_objs** オプションの値を変更しないでください。この値は、RGW ノードをデプロイする前にのみ変更する必要があります。

関連情報

- Ceph RGW: GC の調整オプション
- RGW 一般設定
- 設定リファレンス

8.13. CEPH OBJECT GATEWAY のデータオブジェクトストレージの最適化

バケットライフサイクルの設定は、データオブジェクトストレージを最適化して効率性を高め、データの存続期間を通じて効果的なストレージを提供します。

Ceph Object Gateway の S3 API は、現在 AWS バケットライフサイクルの設定アクションのサブセットをサポートします。

- 有効期限
- NoncurrentVersionExpiration
- AbortIncompleteMultipartUpload

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- ストレージクラスター内のすべてのノードへの root レベルのアクセス。

8.13.1. バケットライフサイクルの並列スレッド処理

Ceph Object Gateway では、複数の Ceph Object Gateway インスタンス間でバケットライフサイクルの並列スレッド処理が可能になりました。並行して実行されるスレッドの数を増やすと、Ceph Object Gateway は大きなワークロードをより効率的に処理できるようになります。さらに、Ceph Object Gateway は、順序どおりの番号付けを使用する代わりに、インデックスシャードの列挙に番号付けされたシーケンスを使用するようになりました。

8.13.2. バケットのライフサイクルの最適化

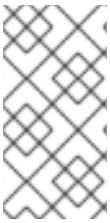
Ceph 設定ファイルにある 2 つのオプションは、バケットのライフサイクル処理の効率性に影響を与えます。

- **rgw_lc_max_worker** は、並行して実行するライフサイクルワーカースレッドの数を指定します。これにより、バケットシャードとインデックスシャードの両方を同時に処理できます。このオプションのデフォルト値は 3 です。

- **rgw_lc_max_wp_worker** は、各ライフサイクルワーカーのワーカースレッドプールのスレッド数を指定します。このオプションを使用すると、各バケットの処理を加速することができます。このオプションのデフォルト値は 3 です。

バケット数が多いワークロード (たとえば、バケット数が数千のワークロード) の場合は、**rgw_lc_max_worker** オプションの値を増やすことを検討してください。

バケットの数は少なく、各バケットのオブジェクトの数が多い (数十万など) ワークロードの場合は、**rgw_lc_max_wp_worker** オプションの値を増やすことを検討してください。



注記

これらのオプションのいずれかの値を増やす前に、現在のストレージクラスターのパフォーマンスと Ceph Object Gateway 使用率を検証してください。Red Hat では、これらのオプションのいずれかに対して、10 以上の値を割り当てることは推奨していません。

前提条件

- 稼働中の Red Hat Ceph Storage クラスターがある。
- ストレージクラスター内のすべてのノードへの root レベルのアクセス。

手順

1. 並行して実行するスレッドの数を増やすには、**rgw_lc_max_worker** の値を **3** から **9** までの値に設定します。

   例

   ```
   [ceph: root@host01 /]# ceph config set client.rgw rgw_lc_max_worker 7
   ```

2. 各スレッドのワークプールのスレッド数を増やすには、**rgw_lc_max_wp_worker** の値を **3** から **9** までの値に設定します。

   例

   ```
   [ceph: root@host01 /]# ceph config set client.rgw rgw_lc_max_wp_worker 7
   ```

3. Ceph Object Gateway を再起動して、変更した設定が有効になるようにします。
4. ストレージクラスターを監視して、値を増やしてもパフォーマンスに悪影響がないことを確認します。

関連情報

- バケットのライフサイクルと並列スレッド処理の詳細は、**バケットのライフサイクルの並列処理** を参照してください。
- Ceph Object Gateway のライフサイクルについての詳細は、**Red Hat サポート** にお問い合わせください。

8.14. AMAZON S3 クラウドサービスへのデータの移行

ストレージクラスを使用してライフサイクル設定の一部としてデータをリモートクラウドサービスに移行し、コストを削減して管理性を向上させることができます。移行は単方向であり、リモートゾーンからデータを戻すことはできません。Amazon (S3) など複数のクラウドプロバイダーへのデータ移行を可能にする機能です。

cloud-s3 を **tier-type** として使用して、データの移行先となるリモートクラウド S3 オブジェクトストアサービスを設定します。これらはデータプールを必要とせず、ゾーングループ配置ターゲットに関して定義されます。

前提条件

- Ceph Object Gateway がインストールされた Red Hat Ceph Storage クラスター。
- リモートクラウドサービス、Amazon S3 のユーザー認証情報。
- Amazon S3 で作成されたターゲットパス。
- ブートストラップされたノードにインストールされた **s3cmd**。
- データをダウンロードするようにローカルに設定された Amazon AWS。

手順

1. アクセスキーとシークレットキーを使用してユーザーを作成します。

   構文

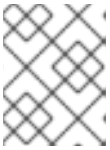
   ```
   radosgw-admin user create --uid=USER_NAME --display-name="DISPLAY_NAME" [--access-key ACCESS_KEY --secret-key SECRET_KEY]
   ```

   例

   ```
   [ceph: root@host01 /]# radosgw-admin user create --uid=test-user --display-name="test-user" --access-key a21e86bce636c3aa1 --secret-key cf764951f1fdde5e
   {
       "user_id": "test-user",
       "display_name": "test-user",
       "email": "",
       "suspended": 0,
       "max_buckets": 1000,
       "subusers": [],
       "keys": [
           {
               "user": "test-user",
               "access_key": "a21e86bce636c3aa1",
               "secret_key": "cf764951f1fdde5e"
           }
       ],
       "swift_keys": [],
       "caps": [],
       "op_mask": "read, write, delete",
       "default_placement": "",
       "default_storage_class": "",
       "placement_tags": [],
       "bucket_quota": {
   ```

```
        "enabled": false,
        "check_on_raw": false,
        "max_size": -1,
        "max_size_kb": 0,
        "max_objects": -1
    },
    "user_quota": {
        "enabled": false,
        "check_on_raw": false,
        "max_size": -1,
        "max_size_kb": 0,
        "max_objects": -1
    },
    "temp_url_keys": [],
    "type": "rgw",
    "mfa_ids": []
}
```

2. ブートストラップされたノードで、階層タイプが **cloud-s3** のストレージクラスを追加します。



注記

--tier-type=cloud-s3 オプションを使用してストレージクラスを作成すると、後で他のストレージクラスターイプに変更することはできません。

構文

```
radosgw-admin zonegroup placement add --rgw-zonegroup =ZONE_GROUP_NAME \
                      --placement-id=PLACEMENT_ID \
                      --storage-class =STORAGE_CLASS_NAME \
                      --tier-type=cloud-s3
```

例

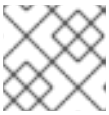
```
[ceph: root@host01 /]# radosgw-admin zonegroup placement add --rgw-zonegroup=default \
                                    --placement-id=default-placement \
                                    --storage-class=CLOUDTIER \
                                    --tier-type=cloud-s3
[
    {
        "key": "default-placement",
        "val": {
            "name": "default-placement",
            "tags": [],
            "storage_classes": [
                "CLOUDTIER",
                "STANDARD"
            ],
            "tier_targets": [
                {
                    "key": "CLOUDTIER",
                    "val": {
                        "tier_type": "cloud-s3",
                        "storage_class": "CLOUDTIER",
```

```
                "retain_head_object": "false",
                "s3": {
                    "endpoint": "",
                    "access_key": "",
                    "secret": "",
                    "host_style": "path",
                    "target_storage_class": "",
                    "target_path": "",
                    "acl_mappings": [],
                    "multipart_sync_threshold": 33554432,
                    "multipart_min_part_size": 33554432
                }
            }
        }
    ]
  }
]
```

3. **storage_class** を更新します。

注記

クラスターがマルチサイトセットアップの一部である場合は、**period update --commit** を実行して、ゾーングループの変更がマルチサイト内のすべてのゾーンに伝播されるようにします。



注記

access_key と **Secret が** 数字で始まらないようにしてください。

必須パラメーターは次のとおりです。

- **access_key** は、特定の接続に使用されるリモートクラウド S3 アクセスキーです。
- **Secret** は、リモートクラウド S3 サービスの秘密キーです。
- **endpoint** は、リモートクラウド S3 サービスエンドポイントの URL です。
- **region** (AWS の場合) は、リモートクラウド S3 サービスのリージョン名です。

オプションのパラメーターは次のとおりです。

- **target_path** は、ターゲットパスの作成方法を定義します。ターゲットパスは、ソース **bucket-name/object-name** が追加される接頭辞を指定します。接頭辞を指定しない場合、作成される target_path は **rgwx-ZONE_GROUP_NAME-STORAGE_CLASS_NAME-cloud-bucket** です。
- **target_storage_class** は、オブジェクトの遷移先となるターゲットストレージクラスを定義します。接頭辞を指定しない場合、オブジェクトは STANDARD ストレージクラスに移行されます。
- **continue_head_object** が true の場合、クラウドに移行されたオブジェクトのメタデータが保持されます。false (デフォルト) の場合、オブジェクトは遷移後に削除されます。このオプションは、現在のバージョン管理されたオブジェクトでは無視されます。

- **multipart_sync_threshold**は、このサイズ以上のオブジェクトがマルチパートアップロードを使用してクラウドに移行されることを指定します。
- **multipart_min_part_size** は、マルチパートアップロードを使用してオブジェクトを移行するときに使用する最小パートサイズを指定します。

構文

```
radosgw-admin zonegroup placement modify --rgw-zonegroup ZONE_GROUP_NAME \
                                --placement-id PLACEMENT_ID \
                                --storage-class STORAGE_CLASS_NAME  \
                                --tier-config=endpoint=AWS_ENDPOINT_URL,\

access_key=AWS_ACCESS_KEY,secret=AWS_SECRET_KEY,\
                                target_path="TARGET_BUCKET_ON_AWS",\
                                multipart_sync_threshold=44432,\
                                multipart_min_part_size=44432,\
                                retain_head_object=true
                                 region=REGION_NAME
```

例

```
[ceph: root@host01 /]# radosgw-admin zonegroup placement modify --rgw-zonegroup
default
                                                --placement-id default-placement \
                                                --storage-class CLOUDTIER \
                                                --tier-config=endpoint=http://10.0.210.010:8080,\

access_key=a21e86bce636c3aa2,secret=cf764951f1fdde5f,\
                                                target_path="dfqe-bucket-01",\
                                                multipart_sync_threshold=44432,\
                                                multipart_min_part_size=44432,\
                                                retain_head_object=true
                                                region=us-east-1

[
    {
        "key": "default-placement",
        "val": {
            "name": "default-placement",
            "tags": [],
            "storage_classes": [
                "CLOUDTIER",
                "STANDARD",
                "cold.test",
                "hot.test"
            ],
            "tier_targets": [
                {
                    "key": "CLOUDTIER",
                    "val": {
                        "tier_type": "cloud-s3",
                        "storage_class": "CLOUDTIER",
                        "retain_head_object": "true",
                        "s3": {
```

```
                    "endpoint": "http://10.0.210.010:8080",
                    "access_key": "a21e86bce636c3aa2",
                    "secret": "cf764951f1fdde5f",
                    "region": "",
                    "host_style": "path",
                    "target_storage_class": "",
                    "target_path": "dfqe-bucket-01",
                    "acl_mappings": [],
                    "multipart_sync_threshold": 44432,
                    "multipart_min_part_size": 44432
                }
            }
        }
    ]
  }
]
]
```

4. Ceph Object Gateway を再起動します。

構文

```
ceph orch restart CEPH_OBJECT_GATEWAY_SERVICE_NAME
```

例

```
[ceph: root@host 01 /]# ceph orch restart rgw.rgw.1

Scheduled to restart rgw.rgw.1.host03.vkfldf on host 'host03'
```

5. シェルを終了し、root ユーザーとして、ブートストラップされたノードで Amazon S3 を設定します。

例

```
[root@host01 ~]# s3cmd --configure

Enter new values or accept defaults in brackets with Enter.
Refer to user manual for detailed description of all options.

Access key and Secret key are your identifiers for Amazon S3. Leave them empty for using the env variables.
Access Key: a21e86bce636c3aa2
Secret Key: cf764951f1fdde5f
Default Region [US]:

Use "s3.amazonaws.com" for S3 Endpoint and not modify it to the target Amazon S3.
S3 Endpoint [s3.amazonaws.com]: 10.0.210.78:80

Use "%(bucket)s.s3.amazonaws.com" to the target Amazon S3. "%(bucket)s" and "%(location)s" vars can be used
if the target S3 system supports dns based buckets.
DNS-style bucket+hostname:port template for accessing a bucket [%
```

```
(bucket)s.s3.amazonaws.com]: 10.0.210.78:80

Encryption password is used to protect your files from reading
by unauthorized persons while in transfer to S3
Encryption password:
Path to GPG program [/usr/bin/gpg]:

When using secure HTTPS protocol all communication with Amazon S3
servers is protected from 3rd party eavesdropping. This method is
slower than plain HTTP, and can only be proxied with Python 2.7 or newer
Use HTTPS protocol [Yes]: No

On some networks all internet access must go through a HTTP proxy.
Try setting it here if you can't connect to S3 directly
HTTP Proxy server name:

New settings:
  Access Key: a21e86bce636c3aa2
  Secret Key: cf764951f1fdde5f
  Default Region: US
  S3 Endpoint: 10.0.210.78:80
  DNS-style bucket+hostname:port template for accessing a bucket: 10.0.210.78:80
  Encryption password:
  Path to GPG program: /usr/bin/gpg
  Use HTTPS protocol: False
  HTTP Proxy server name:
  HTTP Proxy server port: 0

Test access with supplied credentials? [Y/n] Y
Please wait, attempting to list all buckets...
Success. Your access key and secret key worked fine :-)

Now verifying that encryption works...
Not configured. Never mind.

Save settings? [y/N] y
Configuration saved to '/root/.s3cfg'
```

6. S3 バケットを作成します。

構文

```
s3cmd mb s3://NAME_OF_THE_BUCKET_FOR_S3
```

例

```
[root@host01 ~]# s3cmd mb s3://awstestbucket
Bucket 's3://awstestbucket/' created
```

7. ファイルを作成し、すべてのデータを入力して、S3 サービスに移動します。

構文

```
s3cmd put FILE_NAME  s3://NAME_OF_THE_BUCKET_ON_S3
```

例

```
[root@host01 ~]# s3cmd put test.txt s3://awstestbucket

upload: 'test.txt' -> 's3://awstestbucket/test.txt'  [1 of 1]
 21 of 21   100% in    1s    16.75 B/s  done
```

8. ライフサイクル設定移行ポリシーを作成します。

構文

```
<LifecycleConfiguration>
  <Rule>
    <ID>RULE_NAME</ID>
    <Filter>
      <Prefix></Prefix>
    </Filter>
    <Status>Enabled</Status>
    <Transition>
      <Days>DAYS</Days>
      <StorageClass>STORAGE_CLASS_NAME</StorageClass>
    </Transition>
  </Rule>
</LifecycleConfiguration>
```

例

```
[root@host01 ~]# cat lc_cloud.xml
<LifecycleConfiguration>
  <Rule>
    <ID>Archive all objects</ID>
    <Filter>
      <Prefix></Prefix>
    </Filter>
    <Status>Enabled</Status>
    <Transition>
      <Days>2</Days>
      <StorageClass>CLOUDTIER</StorageClass>
    </Transition>
  </Rule>
</LifecycleConfiguration>
```

9. ライフサイクル設定移行ポリシーを設定します。

構文

```
s3cmd setlifecycle FILE_NAME s3://NAME_OF_THE_BUCKET_FOR_S3
```

例

```
[root@host01 ~]#  s3cmd setlifecycle lc_config.xml s3://awstestbucket

s3://awstestbucket/: Lifecycle Policy updated
```

10. **cephadm shell** にログインします。

    例

    ```
    [root@host 01 ~]# cephadm shell
    ```

11. Ceph Object Gateway を再起動します。

    構文

    ```
    ceph orch restart CEPH_OBJECT_GATEWAY_SERVICE_NAME
    ```

    例

    ```
    [ceph: root@host 01 /]# ceph orch restart rgw.rgw.1

    Scheduled to restart rgw.rgw.1.host03.vkfldf on host 'host03'
    ```

検証

1. ソースクラスターで、**radosgw-admin lc list** コマンドを使用して、データが S3 に移動したかどうかを確認します。

    例

    ```
    [ceph: root@host01 /]# radosgw-admin lc list
    [
        {
            "bucket": ":awstestbucket:552a3adb-39e0-40f6-8c84-00590ed70097.54639.1",
            "started": "Mon, 26 Sep 2022 18:32:07 GMT",
            "status": "COMPLETE"
        }
    ]
    ```

2. クラウドエンドポイントでのオブジェクトの移行を確認します。

    例

    ```
    [root@client ~]$ radosgw-admin bucket list
    [
        "awstestbucket"
    ]
    ```

3. バケット内のオブジェクトを一覧表示します。

    例

    ```
    [root@host01 ~]$ aws s3api list-objects --bucket awstestbucket --
    endpoint=http://10.0.209.002:8080
    {
        "Contents": [
            {
                "Key": "awstestbucket/test",
    ```

```
            "LastModified": "2022-08-25T16:14:23.118Z",
            "ETag": "\"378c905939cc4459d249662dfae9fd6f\"",
            "Size": 29,
            "StorageClass": "STANDARD",
            "Owner": {
                "DisplayName": "test-user",
                "ID": "test-user"
            }
        }
    ]
}
```

4. S3 バケットの内容を一覧表示します。

   例

```
[root@host01 ~]# s3cmd ls s3://awstestbucket
2022-08-25 09:57            0  s3://awstestbucket/test.txt
```

5. ファイルの情報を確認します。

   例

```
[root@host01 ~]# s3cmd info s3://awstestbucket/test.txt
s3://awstestbucket/test.txt (object):
   File size: 0
   Last mod:  Mon, 03 Aug 2022 09:57:49 GMT
   MIME type: text/plain
   Storage:   CLOUDTIER
   MD5 sum:   991d2528bb41bb839d1a9ed74b710794
   SSE:       none
   Policy:    none
   CORS:      none
   ACL:       test-user: FULL_CONTROL
   x-amz-meta-s3cmd-attrs:
atime:1664790668/ctime:1664790668/gid:0/gname:root/md5:991d2528bb41bb839d1a9ed74b7
10794/mode:33188/mtime:1664790668/uid:0/uname:root
```

6. Amazon S3 からローカルにデータをダウンロードします。

   a. AWS を設定します。

      例

```
[client@client01 ~]$ aws configure

AWS Access Key ID [****************6VVP]:
AWS Secret Access Key [****************pXqy]:
Default region name [us-east-1]:
Default output format [json]:
```

   b. AWS バケットの内容を一覧表示します。

      例

```
[client@client01 ~]$ aws s3 ls s3://dfqe-bucket-01/awstest
PRE awstestbucket/
```

c. S3 からデータをダウンロードします。

例

```
[client@client01 ~]$ aws s3 cp s3://dfqe-bucket-01/awstestbucket/test.txt .

download: s3://dfqe-bucket-01/awstestbucket/test.txt to ./test.txt
```

8.15. AZURE クラウドサービスへのデータの移行

ストレージクラスを使用してライフサイクル設定の一部としてデータをリモートクラウドサービスに移行し、コストを削減して管理性を向上させることができます。移行は単方向であり、リモートゾーンからデータを戻すことはできません。この機能は、Azure などの複数のクラウドプロバイダーへのデータ移行を可能にするものです。AWS 設定との主な違いの 1 つは、マルチクラウドゲートウェイ (MCG) を設定し、MCG を使用して S3 プロトコルから Azure Blob に変換する必要があることです。

cloud-s3 を **tier-type** として使用して、データの移行先となるリモートクラウド S3 オブジェクトストアサービスを設定します。これらはデータプールを必要とせず、ゾーングループ配置ターゲットに関して定義されます。

前提条件

- Ceph Object Gateway がインストールされた Red Hat Ceph Storage クラスター。
- リモートクラウドサービス Azure のユーザー認証情報。
- Azure はデータをダウンロードするようにローカルに設定されています。
- ブートストラップされたノードにインストールされた **s3cmd**。
- MCG 名前空間用の Azure コンテナーが作成されました。この例では、**mcgnamespace** です。

手順

1. アクセスキーとシークレットキーを使用してユーザーを作成します。

構文

```
radosgw-admin user create --uid=USER_NAME --display-name="DISPLAY_NAME" [--
access-key ACCESS_KEY --secret-key SECRET_KEY]
```

例

```
[ceph: root@host01 /]# radosgw-admin user create --uid=test-user --display-name="test-
user" --access-key a21e86bce636c3aa1 --secret-key cf764951f1fdde5e
{
    "user_id": "test-user",
    "display_name": "test-user",
    "email": "",
    "suspended": 0,
```

```
    "max_buckets": 1000,
    "subusers": [],
    "keys": [
        {
            "user": "test-user",
            "access_key": "a21e86bce636c3aa1",
            "secret_key": "cf764951f1fdde5e"
        }
    ],
    "swift_keys": [],
    "caps": [],
    "op_mask": "read, write, delete",
    "default_placement": "",
    "default_storage_class": "",
    "placement_tags": [],
    "bucket_quota": {
        "enabled": false,
        "check_on_raw": false,
        "max_size": -1,
        "max_size_kb": 0,
        "max_objects": -1
    },
    "user_quota": {
        "enabled": false,
        "check_on_raw": false,
        "max_size": -1,
        "max_size_kb": 0,
        "max_objects": -1
    },
    "temp_url_keys": [],
    "type": "rgw",
    "mfa_ids": []
}
```

2. root ユーザーとして、ユーザー認証情報を使用して AWS CLI を設定し、デフォルトの配置でバケットを作成します。

   構文

   ```
   aws s3 --ca-bundle CA_PERMISSION --profile rgw  --endpoint ENDPOINT_URL --region default mb s3://BUCKET_NAME
   ```

   例

   ```
   [root@host01 ~]$ aws s3   --ca-bundle  /etc/pki/ca-trust/source/anchors/myCA.pem --profile rgw  --endpoint https://host02.example.com:8043 --region default mb s3://transition
   ```

3. バケットが配置ルールで **default-placement** を使用していることを確認します。

   例

   ```
   [root@host01 ~]# radosgw-admin bucket stats --bucket transition
   {
       "bucket": "transition",
   ```

```
    "num_shards": 11,
    "tenant": "",
    "zonegroup": "b29b0e50-1301-4330-99fc-5cdcfc349acf",
    "placement_rule": "default-placement",
    "explicit_placement": {
        "data_pool": "",
        "data_extra_pool": "",
        "index_pool": ""
    },
```

4. OpenShift Data Foundation (ODF) がデプロイされた OpenShift Container Platform (OCP) クラスターにログインします。

   例

```
[root@host01 ~]$ oc project openshift-storage
[root@host01 ~]$ oc get clusterversion
NAME     VERSION   AVAILABLE   PROGRESSING   SINCE   STATUS
version   4.11.6    True        False         4d1h    Cluster version is 4.11.6

[root@host01 ~]$ oc get storagecluster
NAME                 AGE   PHASE   EXTERNAL   CREATED AT            VERSION
ocs-storagecluster   4d    Ready              2023-06-27T15:23:01Z   4.11.0
```

5. Azure の OCP クラスター上で実行されるマルチクラウドゲートウェイ (MCG) 名前空間 Azure バケットを設定します。

   構文

```
noobaa namespacestore create azure-blob az --account-key='ACCOUNT_KEY' --account-name='ACCOUNT_NAME' --target-blob-container='_AZURE_CONTAINER_NAME'
```

   例

```
[root@host01 ~]$ noobaa namespacestore create azure-blob  az --account-key='iq3+6hRtt9bQ46QfHKQ0nSm2aP+tyMzdn8dBSRW4XWrFhY+1nwfqEj4hk2q66nmD85E/o5OrrUqo+AStkKwm9w==' --account-name='transitionrgw' --target-blob-container='mcgnamespace'
```

6. **namespacestore** を指す MCG バケットクラスを作成します。

   例

```
[root@host01 ~]$ noobaa bucketclass create namespace-bucketclass single aznamespace-bucket-class --resource az -n openshift-storage
```

7. クラウドへの移行のためのオブジェクトバケットクレーム (OBC) を作成します。

   構文

```
noobaa obc create OBC_NAME --bucketclass aznamespace-bucket-class -n openshift-storage
```

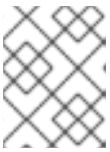
例

```
[root@host01 ~]$ noobaa obc create rgwobc --bucketclass aznamespace-bucket-class -n
openshift-storage
```

注記

OBC によって提供される認証情報を使用して、Ceph Object Gateway でのゾーングループの配置を設定します。

8. ブートストラップされたノードで、Azure で以前に設定された MCG 上のデフォルトのゾーングループ内のデフォルトの配置に、層タイプを **cloud-s3** としてストレージクラスを作成します。



注記

--tier-type=cloud-s3 オプションを使用してストレージクラスを作成すると、後で他のストレージクラスターイプに変更することはできません。

構文

```
radosgw-admin zonegroup placement add --rgw-zonegroup =ZONE_GROUP_NAME \
                        --placement-id=PLACEMENT_ID \
                        --storage-class =STORAGE_CLASS_NAME \
                        --tier-type=cloud-s3
```

例

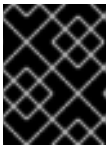
```
[ceph: root@host01 /]# radosgw-admin zonegroup placement add --rgw-zonegroup=default \
                                        --placement-id=default-placement \
                                        --storage-class=AZURE \
                                        --tier-type=cloud-s3
[
   {
      "key": "default-placement",
      "val": {
         "name": "default-placement",
         "tags": [],
         "storage_classes": [
            "AZURE",
            "STANDARD"
         ],
         "tier_targets": [
            {
               "key": "AZURE",
               "val": {
                  "tier_type": "cloud-s3",
                  "storage_class": "AZURE",
                  "retain_head_object": "false",
                  "s3": {
                     "endpoint": "",
                     "access_key": "",
                     "secret": "",
```

```
                    "host_style": "path",
                    "target_storage_class": "",
                    "target_path": "",
                    "acl_mappings": [],
                    "multipart_sync_threshold": 33554432,
                    "multipart_min_part_size": 33554432
                }
            }
        }
    ]
  }
]
```

9. クラウド S3 クラウドストレージクラスを設定します。

構文

```
radosgw-admin zonegroup placement modify --rgw-zonegroup ZONE_GROUP_NAME \
                                         --placement-id PLACEMENT_ID \
                                         --storage-class STORAGE_CLASS_NAME  \
                                         --tier-config=endpoint=ENDPOINT_URL,\
                                         access_key=ACCESS_KEY,secret=SECRET_KEY,\
                                         target_path="TARGET_BUCKET_ON",\
                                         multipart_sync_threshold=44432,\
                                         multipart_min_part_size=44432,\
                                         retain_head_object=true
                                         region=REGION_NAME
```



重要

restart_head_object パラメーターを **true** に設定すると、メタデータまたはオブジェクトのヘッドが保持され、遷移されるオブジェクトがリストされます。

例

```
[ceph: root@host01 /]# radosgw-admin zonegroup placement modify --rgw-zonegroup default
                                                --placement-id default-placement \
                                                --storage-class AZURE \
                                                --tier-config=endpoint="https://s3-openshift-
storage.apps.ocp410.0e73azopenshift.com",\

access_key=a21e86bce636c3aa2,secret=cf764951f1fdde5f,\
                                                target_path="dfqe-bucket-01",\
                                                multipart_sync_threshold=44432,\
                                                multipart_min_part_size=44432,\
                                                retain_head_object=true
                                                region=us-east-1

[
   {
      "key": "default-placement",
      "val": {
         "name": "default-placement",
```

```
            "tags": [],
            "storage_classes": [
                "AZURE",
                "STANDARD",
                "cold.test",
                "hot.test"
            ],
            "tier_targets": [
                {
                    "key": "AZURE",
                    "val": {
                        "tier_type": "cloud-s3",
                        "storage_class": "AZURE",
                        "retain_head_object": "true",
                        "s3": {
                            "endpoint": "https://s3-openshift-
storage.apps.ocp410.0e73azopenshift.com",
                            "access_key": "a21e86bce636c3aa2",
                            "secret": "cf764951f1fdde5f",
                            "region": "",
                            "host_style": "path",
                            "target_storage_class": "",
                            "target_path": "dfqe-bucket-01",
                            "acl_mappings": [],
                            "multipart_sync_threshold": 44432,
                            "multipart_min_part_size": 44432
                        }
                    }
                }
            ]
        }
    }
]
]
```

10. Ceph Object Gateway を再起動します。

構文

```
ceph orch restart CEPH_OBJECT_GATEWAY_SERVICE_NAME
```

例

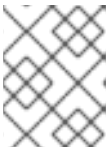
```
[ceph: root@host 01 /]# ceph orch restart client.rgw.objectgwhttps.host02.udyllp

Scheduled to restart client.rgw.objectgwhttps.host02.udyllp on host 'host02
```

11. 前に作成したバケットのライフサイクル設定移行ポリシーを作成します。この例では、バケットは **transition** です。

構文

```
cat transition.json
{
```

```
    "Rules": [
        {
            "Filter": {
                "Prefix": ""
            },
            "Status": "Enabled",
            "Transitions": [
                {
                    "Days": 30,
                    "StorageClass": "STORAGE_CLASS"
                }
            ],
            "ID": "TRANSITION_ID"
        }
    ]
        }
```



注記

30 日以上経過したバケット内のすべてのオブジェクトは、**AZURE** というクラウドストレージクラスに転送されます。

例

```
[root@host01 ~]$ cat transition.json
{
    "Rules": [
        {
            "Filter": {
                "Prefix": ""
            },
            "Status": "Enabled",
            "Transitions": [
                {
                    "Days": 30,
                    "StorageClass": "AZURE"
                }
            ],
            "ID": "Transition Objects in bucket to AZURE Blob after 30 days"
        }
    ]
        }
```

12. AWS CLI を使用してバケットのライフサイクル設定を適用します。

構文

```
 aws s3api --ca-bundle CA_PERMISSION --profile rgw --endpoint ENDPOINT_URL--region default put-bucket-lifecycle-configuration --lifecycle-configuration  file://BUCKET.json --bucket BUCKET_NAME
```

例

```
[root@host01 ~]$ aws s3api   --ca-bundle  /etc/pki/ca-trust/source/anchors/myCA.pem --profile rgw  --endpoint https://host02.example.com:8043 --region default put-bucket-lifecycle-configuration --lifecycle-configuration  file://transition.json --bucket transition
```

13. オプション: ライフサイクル設定を取得します。

構文

```
 aws s3api --ca-bundle CA_PERMISSION --profile rgw --endpoint ENDPOINT_URL--region default get-bucket-lifecycle-configuration --lifecycle-configuration  file://BUCKET.json --bucket BUCKET_NAME
```

例

```
[root@host01 ~]$ aws s3api   --ca-bundle  /etc/pki/ca-trust/source/anchors/myCA.pem --profile rgw  --endpoint https://host02.example.com:8043 --region default get-bucket-lifecycle-configuration --bucket transition
{
    "Rules": [
        {
            "ID": "Transition Objects in bucket to AZURE Blob after 30 days",
            "Prefix": "",
            "Status": "Enabled",
            "Transitions": [
                {
                    "Days": 30,
                    "StorageClass": "AZURE"
                }
            ]
        }
    ]
}
```

14. オプション: **radosgw-admin lc list** コマンドを使用してライフサイクル設定を取得します。

例

```
[root@host 01 ~]# radosgw-admin lc list
[
    {
        "bucket": ":transition:d9c4f708-5598-4c44-9d36-849552a08c4d.169377.1",
        "started": "Thu, 01 Jan 1970 00:00:00 GMT",
        "status": "UNINITIAL"
    }
]
```

注記

UNINITAL ステータスは、ライフサイクル設定が処理されていないことを意味します。移行プロセスが完了すると、**COMPLETED** 状態に移行します。

15. **cephadm shell** にログインします。

例

例

```
[root@host 01 ~]# cephadm shell
```

16. Ceph Object Gateway デーモンを再起動します。

構文

```
ceph orch daemon CEPH_OBJECT_GATEWAY_DAEMON_NAME
```

例

```
[ceph: root@host 01 /]# ceph orch daemon restart rgw.objectgwhttps.host02.udyllp
[ceph: root@host 01 /]# ceph orch daemon restart rgw.objectgw.host02.afwvyq
[ceph: root@host 01 /]# ceph orch daemon restart rgw.objectgw.host05.ucpsrr
```

17. ソースクラスターから Azure にデータを移行します。

例

```
[root@host 01 ~]# for i in 1 2 3 4 5
do
aws s3 --ca-bundle  /etc/pki/ca-trust/source/anchors/myCA.pem --profile rgw --endpoint
https://host02.example.com:8043 --region default cp /etc/hosts s3://transition/transition$i
done
```

18. データの移行を確認します。

例

```
[root@host 01 ~]# aws s3 --ca-bundle  /etc/pki/ca-trust/source/anchors/myCA.pem --profile
rgw  --endpoint https://host02.example.com:8043 --region default ls s3://transition
2023-06-30 10:24:01       3847 transition1
2023-06-30 10:24:04       3847 transition2
2023-06-30 10:24:07       3847 transition3
2023-06-30 10:24:09       3847 transition4
2023-06-30 10:24:13       3847 transition5
```

19. **rados ls** コマンドを使用して、データが Azure に移動したかどうかを確認します。

例

```
[root@host 01 ~]# rados ls -p default.rgw.buckets.data | grep transition
d9c4f708-5598-4c44-9d36-849552a08c4d.169377.1_transition1
d9c4f708-5598-4c44-9d36-849552a08c4d.169377.1_transition4
d9c4f708-5598-4c44-9d36-849552a08c4d.169377.1_transition2
d9c4f708-5598-4c44-9d36-849552a08c4d.169377.1_transition3
d9c4f708-5598-4c44-9d36-849552a08c4d.169377.1_transition5
```

20. データが移行されていない場合は、**lc process** コマンドを実行できます。

例

```
[root@host 01 ~]# radosgw-admin lc process
```

これにより、ライフサイクルプロセスが強制的に開始され、設定されているすべてのバケットライフサイクルポリシーが評価されます。その後、必要に応じてデータの移行を開始します。

検証

1. **radosgw-admin lc list** コマンドを実行して、移行が完了したことを確認します。

   例

```
[root@host 01 ~]#  radosgw-admin lc list
[
    {
        "bucket": ":transition:d9c4f708-5598-4c44-9d36-849552a08c4d.170017.5",
        "started": "Mon, 30 Jun 2023-06-30 16:52:56 GMT",
        "status": "COMPLETE"
    }
]
```

2. バケット内のオブジェクトを一覧表示します。

   例

```
[root@host01 ~]$ aws s3api list-objects --bucket awstestbucket --
endpoint=http://10.0.209.002:8080
{
    "Contents": [
        {
            "Key": "awstestbucket/test",
            "LastModified": "2023-06-25T16:14:23.118Z",
            "ETag": "\"378c905939cc4459d249662dfae9fd6f\"",
            "Size": 29,
            "StorageClass": "STANDARD",
            "Owner": {
                "DisplayName": "test-user",
                "ID": "test-user"
            }
        }
    ]
}
```

3. クラスター上のオブジェクトをリストします。

   例

```
[root@host01 ~]$ aws s3 --ca-bundle  /etc/pki/ca-trust/source/anchors/myCA.pem --profile
rgw  --endpoint https://host02.example.com:8043 --region default ls s3://transition
2023-06-30 17:52:56          0 transition1
2023-06-30 17:51:59          0 transition2
2023-06-30 17:51:59          0 transition3
2023-06-30 17:51:58          0 transition4
2023-06-30 17:51:59          0 transition5
```

オブジェクトのサイズは **0** です。オブジェクトをリスト表示することはできますが、オブジェクトは Azure に移行されているため、コピーすることはできません。

4. S3 API を使用してオブジェクトの先頭を確認します。

例

```
[root@host01 ~]$ aws s3api --ca-bundle  /etc/pki/ca-trust/source/anchors/myCA.pem --profile
rgw  --endpoint https://host02.example.com:8043 --region default head-object --key
transition1 --bucket transition
{
    "AcceptRanges": "bytes",
    "LastModified": "2023-06-31T16:52:56+00:00",
    "ContentLength": 0,
    "ETag": "\"46ecb42fd0def0e42f85922d62d06766\"",
    "ContentType": "binary/octet-stream",
    "Metadata": {},
    "StorageClass": "CLOUDTIER"
}
```

ストレージクラスが **STANDARD** から **CLOUDTIER** に変更したことがわかります。

第9章 テスト

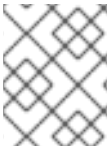
ストレージ管理者は、基本的な機能テストを実行して、Ceph Object Gateway 環境が想定どおりに機能していることを確認できます。S3 インターフェイス用に初期 Ceph Object Gateway ユーザーを作成して、Swift インターフェイスのサブユーザーを作成して、REST インターフェイスを使用できます。

前提条件

- 正常かつ実行中の Red Hat Ceph Storage クラスター
- Ceph Object Gateway ソフトウェアのインストール。

9.1. S3 ユーザーを作成します。

ゲートウェイをテストするには、S3 ユーザーを作成し、ユーザーアクセスを付与します。**man radosgw-admin** コマンドは、追加のコマンドオプションに関する情報を提供します。



注記

マルチサイトのデプロイメントでは、マスターゾーングループのマスターゾーンにあるホストでユーザーを作成します。

前提条件

- **root** または **sudo** アクセス
- Ceph Object Gateway がインストールされていること

手順

1. S3 ユーザーを作成します。

   構文

   ```
   radosgw-admin user create --uid=name --display-name="USER_NAME"
   ```

   name を S3 ユーザーの名前に置き換えます。

   例

   ```
   [root@host01 ~]# radosgw-admin user create --uid="testuser" --display-name="Jane Doe"
   {
       "user_id": "testuser",
       "display_name": "Jane Doe",
       "email": "",
       "suspended": 0,
       "max_buckets": 1000,
       "auid": 0,
       "subusers": [],
       "keys": [
           {
               "user": "testuser",
               "access_key": "CEP28KDIQXBKU4M15PDC",
   ```

```
            "secret_key": "MARoio8HFc8JxhEilES3dKFVj8tV3NOOYymihTLO"
        }
    ],
    "swift_keys": [],
    "caps": [],
    "op_mask": "read, write, delete",
    "default_placement": "",
    "placement_tags": [],
    "bucket_quota": {
        "enabled": false,
        "check_on_raw": false,
        "max_size": -1,
        "max_size_kb": 0,
        "max_objects": -1
    },
    "user_quota": {
        "enabled": false,
        "check_on_raw": false,
        "max_size": -1,
        "max_size_kb": 0,
        "max_objects": -1
    },
    "temp_url_keys": [],
    "type": "rgw"
}
```


2. 出力を確認し、**access_key** および **secret_key** の値に JSON エスケープ文字 (\) が含まれていないことを確認します。これらの値はアクセスの検証に必要ですが、値に JSON エスケープ文字が含まれる場合、特定のクライアントは処理できません。この問題を修正するには、以下のアクションのいずれかを実行します。

   - JSON エスケープ文字を削除します。
   - 文字列を引用符でカプセル化します。
   - キーを再生成し、JSON エスケープ文字が含まれていないことを確認します。
   - キーおよびシークレットを手動で指定します。

   正引きスラッシュ / は有効な文字であるため、削除しないでください。

9.2. SWIFT ユーザーの作成

Swift インターフェイスをテストするには、Swift サブユーザーを作成します。Swift ユーザーは、2 つの手順で作成します。最初のステップでは、ユーザーを作成します。次のステップでは、秘密鍵を作成します。

注記

マルチサイトのデプロイメントでは、マスターゾーングループのマスターゾーンにあるホストでユーザーを作成します。

前提条件

- Ceph Object Gateway のインストール

- Ceph Object Gateway ノードへのルートレベルのアクセスがある。

手順

1. Swift ユーザーを作成します。

   構文

   ```
   radosgw-admin subuser create --uid=NAME --subuser=NAME:swift --access=full
   ```

   NAME を Swift ユーザー名に置き換えます。以下に例を示します。

   例

   ```
   [root@host01 ~]# radosgw-admin subuser create --uid=testuser --subuser=testuser:swift --
   access=full
   {
       "user_id": "testuser",
       "display_name": "First User",
       "email": "",
       "suspended": 0,
       "max_buckets": 1000,
       "auid": 0,
       "subusers": [
           {
               "id": "testuser:swift",
               "permissions": "full-control"
           }
       ],
       "keys": [
           {
               "user": "testuser",
               "access_key": "O8JDE41XMI74O185EHKD",
               "secret_key": "i4Au2yxG5wtr1JK01mI8kjJPM93HNAoVWOSTdJd6"
           }
       ],
       "swift_keys": [
           {
               "user": "testuser:swift",
               "secret_key": "13TLtdEW7bCqgttQgPzxFxziu0AgabtOc6vM8DLA"
           }
       ],
       "caps": [],
       "op_mask": "read, write, delete",
       "default_placement": "",
       "placement_tags": [],
       "bucket_quota": {
           "enabled": false,
           "check_on_raw": false,
           "max_size": -1,
           "max_size_kb": 0,
           "max_objects": -1
       },
       "user_quota": {
           "enabled": false,
   ```

```
        "check_on_raw": false,
        "max_size": -1,
        "max_size_kb": 0,
        "max_objects": -1
    },
    "temp_url_keys": [],
    "type": "rgw"
}
```

2. シークレットキーを作成します。

構文

```
radosgw-admin key create --subuser=NAME:swift --key-type=swift --gen-secret
```

NAME を Swift ユーザー名に置き換えます。以下に例を示します。

例

```
[root@host01 ~]# radosgw-admin key create --subuser=testuser:swift --key-type=swift --gen-
secret
{
    "user_id": "testuser",
    "display_name": "First User",
    "email": "",
    "suspended": 0,
    "max_buckets": 1000,
    "auid": 0,
    "subusers": [
        {
            "id": "testuser:swift",
            "permissions": "full-control"
        }
    ],
    "keys": [
        {
            "user": "testuser",
            "access_key": "O8JDE41XMI74O185EHKD",
            "secret_key": "i4Au2yxG5wtr1JK01mI8kjJPM93HNAoVWOSTdJd6"
        }
    ],
    "swift_keys": [
        {
            "user": "testuser:swift",
            "secret_key": "a4ioT4jEP653CDcdU8p4OuhruwABBRZmyNUbnSSt"
        }
    ],
    "caps": [],
    "op_mask": "read, write, delete",
    "default_placement": "",
    "placement_tags": [],
    "bucket_quota": {
        "enabled": false,
        "check_on_raw": false,
        "max_size": -1,
```

```
        "max_size_kb": 0,
        "max_objects": -1
    },
    "user_quota": {
        "enabled": false,
        "check_on_raw": false,
        "max_size": -1,
        "max_size_kb": 0,
        "max_objects": -1
    },
    "temp_url_keys": [],
    "type": "rgw"
}
```

9.3. S3 アクセスのテスト

S3 アクセスを検証するには、Python テストスクリプトを作成し、実行する必要があります。S3 アクセステスト mp スクリプトは **radosgw** に接続し、新規バケットを作成し、すべてのバケットをリスト表示します。**aws_access_key_id** および **aws_secret_access_key** の値は、**radosgw_admin** コマンドで返される **access_key** および **secret_key** の値から取得されます。

前提条件

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

手順

1. Red Hat Enterprise Linux 9 の High Availability リポジトリーを有効にします。

```
subscription-manager repos --enable=rhel-9-for-x86_64-highavailability-rpms
```

2. **python3-boto3** パッケージをインストールします。

```
dnf install python3-boto3
```

3. Python スクリプトを作成します。

```
vi s3test.py
```

4. ファイルに以下のコンテンツを追加します。

構文

```
import boto3

endpoint = "" # enter the endpoint URL along with the port "http://URL:PORT"

access_key = 'ACCESS'
secret_key = 'SECRET'

s3 = boto3.client(
```

```
        's3',
        endpoint_url=endpoint,
        aws_access_key_id=access_key,
        aws_secret_access_key=secret_key
        )

s3.create_bucket(Bucket='my-new-bucket')

response = s3.list_buckets()
for bucket in response['Buckets']:
    print("{name}\t{created}".format(
  name = bucket['Name'],
  created = bucket['CreationDate']
))
```

a. **endpoint** は、ゲートウェイサービスを設定したホストの URL に置き換えます。つまり、**gateway host** です。**host** の設定が DNS で解決されていることを確認します。**PORT** は、ゲートウェイのポート番号に置き換えます。

b. **ACCESS** および **SECRET** は、Red Hat Ceph Storage Object Gateway Guideの Create an S3 User セクションの **access_key** および **secret_key** の値に置き換えます。

5. スクリプトを実行します。

```
python3 s3test.py
```

出力は以下のようになります。

```
my-new-bucket 2022-05-31T17:09:10.000Z
```

9.4. SWIFT アクセスのテスト

Swift アクセスは、**swift** コマンドラインクライアントを使用して検証できます。**man swift** コマンドは、利用可能なコマンドラインオプションの詳細を提供します。

swift クライアントをインストールするには、以下のコマンドを実行します。

```
sudo yum install python-setuptools
sudo easy_install pip
sudo pip install --upgrade setuptools
sudo pip install --upgrade python-swiftclient
```

swift アクセスをテストするには、以下のコマンドを実行します。

構文

```
# swift -A http://IP_ADDRESS:PORT/auth/1.0 -U testuser:swift -K 'SWIFT_SECRET_KEY' list
```

IP_ADDRESS を、ゲートウェイサーバーのパブリック IP アドレスに置き換え、**SWIFT_SECRET_KEY** を、**swift** ユーザーに対して発行した **radosgw-admin key create** コマンドの出力にある値に置き換えます。**PORT** を Beast で使用するポート番号に置き換えます。ポートを置き換えない場合、デフォルトはポート **80** になります。

以下に例を示します。

```
swift -A http://10.10.143.116:80/auth/1.0 -U testuser:swift -K
'244+fz2gSqoHwR3lYtSbIyomyPHf3i7rgSJrF/IA' list
```

出力は以下のようになります。

```
my-new-bucket
```

付録A 設定の参照

ストレージ管理者は、Ceph Object Gateway にさまざまなオプションを設定できます。このようなオプションにはデフォルト値が含まれます。各オプションを指定しない場合、デフォルト値は自動的に設定されます。

このオプションに特定の値を設定するには、**ceph config set client.rgw OPTION VALUE** コマンドを使用して設定データベースを更新します。

A.1. 一般設定

名前	説明	型	デフォルト
rgw_data	Ceph Object Gateway のデータファイルの場所を設定します。	String	**/var/lib/ceph/radosgw/$cluster-$id**
rgw_enable_apis	指定された API を有効にします。	String	**s3, s3website, swift, swift_auth, admin, sts, iam, notifications**
rgw_cache_enabled	Ceph Object Gateway キャッシュを有効にするかどうか。	Boolean	**true**
rgw_cache_lru_size	Ceph Object Gateway キャッシュのエントリー数。	Integer	**10000**
rgw_socket_path	ドメインソケットのソケットパス。**FastCgiExternalServer** はこのソケットを使用します。ソケットパスを指定しない場合、Ceph Object Gateway は外部サーバーとしては実行しません。ここで指定するパスは、**rgw.conf** ファイルで指定されたパスと同じである必要があります。	String	該当なし
rgw_host	Ceph Object Gateway インスタンスのホスト。IP アドレスまたはホスト名を指定できます。	String	**0.0.0.0**
rgw_port	インスタンスが要求をリッスンするポート。指定されていない場合、Ceph Object Gateway は外部の FastCGI を実行します。	String	なし
rgw_dns_name	提供されるドメインの DNS 名。ゾーングループ内での **hostnames** の設定も参照してください。	String	なし

名前	説明	型	デフォルト
rgw_script_uri	リクエストで設定されていない場合は **SCRIPT_URI** の代替値。	String	なし
rgw_request_uri	リクエストで設定されていない場合は **REQUEST_URI** の代替値。	String	なし
rgw_print_continue	稼働中の場合は **100-continue** を有効にします。	Boolean	**true**
rgw_remote_addr_param	リモートアドレスパラメーター。たとえば、リモートアドレスを含む HTTP フィールド、またはリバースプロキシーが動作している場合は **X-Forwarded-For** アドレス。	String	**REMOTE_ADDR**
rgw_op_thread_timeout	オープンスレッドのタイムアウト (秒単位)。	Integer	600
rgw_op_thread_suicide_timeout	Ceph Object Gateway プロセスが終了するまでの **タイムアウト** (秒単位)。**0** に設定すると無効です。	Integer	**0**
rgw_thread_pool_size	スレッドプールのサイズ。	Integer	512 スレッド
rgw_num_control_oids	異なる **rgw** インスタンス間でキャッシュの同期に使用される通知オブジェクトの数。	Integer	**8**
rgw_init_timeout	Ceph Object Gateway が初期化時に起動するまでの時間 (秒数)。	Integer	**30**
rgw_mime_types_file	MIME タイプのパスおよび場所。オブジェクトタイプの Swift の自動検出に使用されます。	String	**/etc/mime.types**
rgw_gc_max_objs	1 つのガベージコレクションの処理サイクルで、ガベージコレクションによって処理される可能性のあるオブジェクトの最大数。	Integer	**32**
rgw_gc_obj_min_wait	オブジェクトが削除され、ガベージコレクション処理によって処理されるまでの最小待機時間。	Integer	**2 * 3600**

名前	説明	型	デフォルト
rgw_gc_processor_max_time	2 つの連続するガベージコレクション処理サイクルで、1 つが開始された後に 2 つ目が開始されるまでの最大時間。	Integer	**3600**
rgw_gc_processor_period	ガベージコレクション処理のサイクル時間。	Integer	**3600**
rgw_s3 success_create_obj_status	**create-obj** の代替成功ステータス応答。	Integer	**0**
rgw_resolve_cname	**rgw** が要求ホスト名フィールドの DNS CNAME レコードを使用するべきかどうか (ホスト名が **rgw_dns 名** と等しくない場合)。	Boolean	**false**
rgw_object_stripe_size	Ceph Object Gateway オブジェクトのオブジェクトストライプのサイズ。	Integer	**4 << 20**
rgw_extended_http_attrs	オブジェクトに設定できる属性の新規セットを追加します。これらの追加属性は、オブジェクトを配置する際に HTTP ヘッダーフィールドを介して設定できます。設定された場合、オブジェクトで GET/HEAD を実行すると、これらの属性は HTTP フィールドとして返されます。	String	なし。例: "content_foo、content_bar"
rgw_exit_timeout_secs	無条件に終了する前にプロセスを待機する秒数。	Integer	**120**
rgw_get_obj_window_size	単一オブジェクト要求のウィンドウサイズ (バイト単位)。	Integer	**16 << 20**
rgw_get_obj_max_req_size	Ceph Storage Cluster に送信される単一の get 操作の最大リクエストサイズ。	Integer	**4 << 20**
rgw_relaxed_s3_bucket_names	ゾーングループバケットの緩和された S3 バケット名ルールを有効にします。	Boolean	**false**
rgw_list buckets_max_chunk	ユーザーバケットをリスト表示する際に、単一の操作で取得するバケットの最大数。	Integer	**1000**

名前	説明	型	デフォルト
rgw_override_bucket_index_max_shards	バケットインデックスオブジェクトのシャードの数。値が **0** の場合は、シャード化がないことを示します。Red Hat では、バケットのリストのコストを増やすため、大きすぎる値 (**1000** など) を設定することは推奨されません。 この変数は、**[client]** セクションまたは **[global]** セクションで設定して、**radosgw-admin** コマンドに自動的に適用されるようにする必要があります。	Integer	**0**
rgw_curl_wait_timeout_ms	特定の **curl** 呼び出しのタイムアウト (ミリ秒単位)。	Integer	**1000**
rgw_copy_obj_progress	長いコピー操作中にオブジェクトの進行状況を出力できるようにします。	Boolean	**true**
rgw_copy_obj_progress_every_bytes	コピー進行状況出力間の最小バイト。	Integer	**1024 * 1024**
rgw_admin_entry	管理要求 URL のエントリーポイント。	String	**admin**
rgw_content_length_compat	CONTENT_LENGTH と HTTP_CONTENT_LENGTH セットの両方で FCGI 要求の互換性処理を有効にします。	Boolean	**false**
rgw_bucket_default_quota_max_objects	バケットごとのデフォルトのオブジェクトの最大数。他のクォータが指定されていない場合、この値は新規ユーザーに設定されます。既存ユーザーには影響しません。 この変数は、**[client]** セクションまたは **[global]** セクションで設定して、**radosgw-admin** コマンドに自動的に適用されるようにする必要があります。	Integer	**-1**
rgw_bucket_quota_ttl	キャッシュされたクォータ情報が信頼される秒単位の時間。このタイムアウト後、クォータ情報がクラスターから再フェッチされます。	Integer	600

名前	説明	型	デフォルト
rgw_user_quota_bucket_sync_interval	クラスターに同期する前に、バケットクォータ情報が累積される時間 (秒単位)。この間に、他の RGW インスタンスは、このインスタンスでの操作によるバケットクォータ統計の変更を認識しません。	Integer	180
rgw_user_quota_sync_interval	クラスターに同期する前に、ユーザークォータ情報が累積される時間 (秒単位)。この間に、他の RGW インスタンスは、このインスタンスでの操作によるユーザークォータ統計の変更を認識しません。	Integer	3600 * 24
log_meta	ゲートウェイがメタデータ操作をログに記録するかどうかを決定するゾーンパラメーター。	Boolean	**false**
log_data	ゲートウェイがデータ操作をログに記録するかどうかを決定するゾーンパラメーター。	Boolean	**false**
sync_from_all	ゾーンがすべてのゾーングループピアから同期するかどうかを設定または設定解除するための **radosgw-admin** コマンド。	Boolean	**false**

A.2. プールについて

Ceph ゾーンは、一連の Ceph Storage Cluster プールにマッピングします。

手動で作成されたプールと生成されたプール

Ceph Object Gateway のユーザーキーに書き込み機能が含まれる場合、ゲートウェイにはプールを自動的に作成する機能があります。これは、使用開始に便利です。ただし、Ceph Object Storage Cluster は、Ceph 設定ファイルに設定されていない限り、配置グループのデフォルト値を使用します。また、Ceph はデフォルトの CRUSH 階層を使用します。これらの設定は、実稼働システムに適して **いません**。

Ceph Object Gateway のデフォルトゾーンのデフォルトプールには以下が含まれます。

- **.rgw.root**
- **.default.rgw.control**
- **.default.rgw.meta**
- **.default.rgw.log**
- **.default.rgw.buckets.index**

- **.default.rgw.buckets.data**
- **.default.rgw.buckets.non-ec**

Ceph Object Gateway は、ゾーンごとにプールを作成します。プールを手動で作成する場合は、ゾーン名を先頭に追加します。システムプールは、たとえばシステム制御、ロギング、ユーザー情報などに関連するオブジェクトを保存します。慣例により、これらのプール名には、プール名の前にゾーン名が付加されます。

- **.<zone-name>.rgw.control**: コントロールプール。
- **.<zone-name>.log**: ログプールには、すべてのバケット/コンテナーのログおよび create、read、update、および delete などのオブジェクトアクションが含まれます。
- **.<zone-name>.rgw.buckets.index**: このプールはバケットのインデックスを保存します。
- **.<zone-name>.rgw.buckets.data**: このプールはバケットのデータを格納します。
- **.<zone-name>.rgw.meta**: メタデータプールは **user_keys** およびその他の重要なメタデータを保存します。
- **.<zone-name>.meta:users.uid**: ユーザー ID プールには、一意のユーザー ID のマップが含まれます。
- **.<zone-name>.meta:users.keys**: keys プールには、各ユーザー ID のアクセスキーと秘密鍵が含まれます。
- **.<zone-name>.meta:users.email**: メールプールには、ユーザー ID に関連付けられたメールアドレスが含まれます。
- **.<zone-name>.meta:users.swift**: Swift プールには、ユーザー ID の Swift サブユーザー情報が含まれます。

Ceph Object Gateways は、配置プールにバケットインデックス (**index_pool**) およびバケットデータ (**data_pool**) のデータを保存します。これらは重複する可能性があります。つまり、インデックスとデータに同じプールを使用できます。デフォルト配置のインデックスプールは **{zone-name}.rgw.buckets.index** であり、デフォルト配置のデータプールのインデックスプールは **{zone-name}.rgw.buckets** です。

名前	説明	型	デフォルト
rgw_zonegroup_root_pool	すべてのゾーングループ固有の情報を保存するプール。	String	**.rgw.root**
rgw_zone_root_pool	ゾーン固有の情報を保存するプール。	String	**.rgw.root**

A.3. ライフサイクル設定

ストレージ管理者は、Ceph Object Gateway にさまざまなバケットライフサイクルオプションを設定できます。このようなオプションにはデフォルト値が含まれます。各オプションを指定しない場合、デフォルト値は自動的に設定されます。

このオプションに特定の値を設定するには、**ceph config set client.rgw OPTION VALUE** コマンドを使用して設定データベースを更新します。

名前	説明	型	デフォルト
rgw_lc_debug_interval	開発者専用。有効期限のルールを日単位から秒単位に拡大し、ライフサイクルルールをデバッグすることができます。Red Hat は、実稼働クラスターではこのオプションを使用しないことを推奨します。	Integer	**-1**
rgw_lc_lock_max_time	Ceph Object Gateway によって内部で使用されるタイムアウト値。	Integer	**90**
rgw_lc_max_objs	RADOS Gateway 内部ライフサイクル作業キューのシャーディングを制御します。意図的な再シャーディングワークフローの一部としてのみ設定する必要があります。Red Hat では、クラスターのセットアップ後に Red Hat サポートに連絡せずにこの設定を変更することは推奨していません。	Integer	**32**
rgw_lc_max_rules	バケットごとに 1 つのライフサイクル設定ドキュメントに含めるライフサイクルルールの数。Amazon Web Service (AWS) の制限は 1000 ルールです。	Integer	**1000**
rgw_lc_max_worker	バケットおよびインデックスシャードを同時に処理する、並行して実行するライフサイクルワーカースレッドの数。Red Hat は、Red Hat サポートに問い合わせることなく 10 を超える値を設定することは推奨していません。	Integer	**3**
rgw_lc_max_wp_worker	各ライフサイクルワーカースレッドが並行して処理できるバケットの数。Red Hat は、Red Hat サポートに問い合わせることなく 10 を超える値を設定することは推奨していません。	Integer	**3**
rgw_lc_thread_delay	複数のポイントでシャードの処理に注入できる遅延 (ミリ秒単位)。デフォルト値は 0 です。値を 10 から 100 ミリ秒に設定すると、RADOS Gateway インスタンスの CPU 使用率が低下し、飽和状態が観察される場合に、インジェストに対するライフサイクルスレッドのワークロード容量の比率が減少します。	Integer	**0**

A.4. SWIFT 設定

名前	説明	型	デフォルト
rgw_enforce_swift_acls	Swift Access Control List (ACL) 設定を強制します。	Boolean	**true**

名前	説明	型	デフォルト
rgw_swift_token_expiration	Swift トークンの有効期限が切れるまでの時間 (秒単位)。	Integer	**24 * 3600**
rgw_swift_url	Ceph Object Gateway Swift API の URL。	String	なし
rgw_swift_url_prefix	Swift API の URL 接頭辞 (例: **http://fqdn.com/swift**)。	**swift**	該当なし
rgw_swift_auth_url	v1 認証トークンを確認するためのデフォルト URL (内部の Swift 認証を使用しない場合)。	String	なし
rgw_swift_auth_entry	Swift 認証 URL のエントリーポイント。	String	**auth**

A.5. LOGGING SETTINGS

名前	説明	型	デフォルト
debug_rgw_datacache	低レベルの D3N ログは、**debug_rgw_datacache** サブシステムによって有効にできます (**debug_rgw_datacache** = **30** まで)	Integer	**1/5**
rgw_log_nonexistent_bucket	Ceph Object Gateway が、存在しないバケットの要求をログに記録できるようにします。	Boolean	**false**
rgw_log_object_name	オブジェクト名のロギング形式。フォーマット指定子の詳細は、man ページの date を参照してください。	Date	**%Y-%m-%d-%H-%i-%n**
rgw_log_object_name_utc	ログに記録されたオブジェクト名に UTC 時刻が含まれているかどうか。**false** の場合はローカルタイムを使用します。	Boolean	**false**
rgw_usage_max_shards	使用状況ログのシャードの最大数。	Integer	**32**
rgw_usage_max_user_shards	1 人のユーザーの使用状況ログに使用されるシャードの最大数。	Integer	**1**
rgw_enable_ops_log	成功した Ceph Object Gateway 操作ごとにロギングを有効にします。	Boolean	**false**
rgw_enable_usage_log	使用状況ログを有効にします。	Boolean	**false**

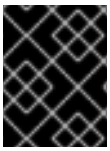
名前	説明	型	デフォルト
rgw_ops_log_rados	操作ログを Ceph Storage Cluster のバックエンドに書き込む必要があるかどうか。	Boolean	**true**
rgw_ops_log_socket_path	操作ログを書き込む Unix ドメインソケット。	String	なし
rgw_ops_log_data-backlog	Unix ドメインソケットに書き込まれた操作ログの最大データバックログデータサイズ。	Integer	**5 << 20**
rgw_usage_log_flush_threshold	同期的にフラッシュする前に、使用状況ログでダーティーマージされたエントリーの数。	Integer	1024
rgw_usage_log_tick_interval	保留中の使用量のログデータを **n** 秒ごとにフラッシュします。	Integer	**30**
rgw_intent_log_object_name	インテントログオブジェクト名のロギング形式。フォーマット指定子の詳細は、man ページの date を参照してください。	Date	**%Y-%m-%d-%i-%n**
rgw_intent_log_object_name_utc	インテントログオブジェクト名に UTC 時刻が含まれているかどうか。**false** の場合はローカルタイムを使用します。	Boolean	**false**
rgw_data_log_window	データログエントリーウィンドウ (秒単位)。	Integer	**30**
rgw_data_log_changes_size	データ変更ログのために保持するインメモリーエントリーの数。	Integer	**1000**
rgw_data_log_num_shards	データ変更ログを保持するシャード (オブジェクト) の数。	Integer	**128**
rgw_data_log_obj_prefix	データログのオブジェクト名の接頭辞。	String	**data_log**
rgw_replica_log_obj_prefix	レプリカログのオブジェクト名の接頭辞。	String	**replica log**
rgw_md_log_max_shards	メタデータログのシャードの最大数。	Integer	**64**
rgw_log_http_headers	ops ログエントリーに含める HTTP ヘッダーのコンマ区切りのリスト。ヘッダー名は大文字と小文字が区別されず、単語がアンダースコアで区切られた完全なヘッダー名を使用します。	String	なし

A.6. KEYSTONE 設定

名前	説明	型	デフォルト
rgw_keystone_url	Keystone サーバーの URL。	String	なし
rgw_keystone_admin_token	Keystone 管理トークン (共有シークレット)	String	なし
rgw_keystone_accepted_roles	要求を提供するのに必要なロール。	String	**Member, admin**
rgw_keystone_token_cache_size	各 Keystone トークンキャッシュのエントリーの最大数。	Integer	**10000**

A.7. KEYSTONE の統合設定オプション

設定オプションは Keystone に統合できます。利用可能な Keystone 統合設定オプションの詳細は、以下を参照してください。



重要

Ceph 設定ファイルを更新したら、新しい Ceph 設定ファイルをストレージクラスター内の全 Ceph ノードにコピーする必要があります。

rgw_s3_auth_use_keystone

説明

true に設定すると、Ceph Object Gateway は Keystone を使用してユーザーを認証します。

型

Boolean

デフォルト

false

nss_db_path

説明

NSS データベースへのパス。

型

String

デフォルト

""

rgw_keystone_url

説明

Keystone サーバーの管理 RESTful API の URL。

型

String

デフォルト

""

rgw_keystone_admin_token

説明

管理リクエストのために Keystone の内部に設定されるトークンまたは共有シークレット。

型

String

デフォルト

""

rgw_keystone_admin_user

説明

keystone 管理ユーザー名

型

String

デフォルト

""

rgw_keystone_admin_password

説明

keystone 管理ユーザーのパスワード。

型

String

デフォルト

""

rgw_keystone_admin_tenant

説明

keystone v2.0 用の Keystone 管理ユーザーテナント。

型

String

デフォルト

""

rgw_keystone_admin_project

説明

keystone v3 の Keystone 管理ユーザープロジェクト。

型

String

デフォルト

""

rgw_trust_forwarded_https

説明

Ceph Object Gateway の前にあるプロキシーが SSL 終了に使用される場合、受信 http 接続が安全であるかどうかは判断されません。接続が安全であるかどうかを判断するときに、プロキシーによって送信された転送ヘッダーと X-forwarded ヘッダーを信頼するには、このオプションを有効にします。これは主にサーバー側の暗号化に必要です。

型

Boolean

デフォルト

false

rgw_swift_account_in_url

説明

Swift アカウントが URL パスでエンコードされているかどうか。Ceph Object Gateway で一般に読み取り可能なコンテナーと一時 URL をサポートする場合は、このオプションを **true** に設定し、Keystone サービスカタログを更新する **必要があります**。

型

Boolean

デフォルト

false

rgw_keystone_admin_domain

説明

Keystone 管理ユーザードメイン。

型

String

デフォルト

""

rgw_keystone_api_version

説明

使用する Keystone API のバージョン。有効なオプションは **2** または **3** です。

型

Integer

デフォルト

2

rgw_keystone_accepted_roles

説明

要求を提供するのに必要なロール。

型

String

デフォルト

member, Member, admin

rgw_keystone_accepted_admin_roles

説明

ユーザーが管理者権限を取得できるようにするロールのリスト。

型

String

デフォルト

ResellerAdmin, swiftoperator

rgw_keystone_token_cache_size

説明

Keystone トークンキャッシュのエントリーの最大数。

型

Integer

デフォルト

10000

rgw_keystone_verify_ssl

説明

true の場合、Ceph は Keystone の SSL 証明書を確認します。

型

Boolean

デフォルト

true

rgw_keystone_implicit_tenants

説明

同じ名前の独自のテナントに新しいユーザーを作成します。ほとんどの場合は、**true** または **false** に設定します。以前のバージョンの Red Hat Ceph Storage との互換性を確保するには、これを **s3** または **swift** に設定することもできます。これにより、ID 領域を分割し、指定されたプロトコルのみが暗黙的なテナントを使用します。Red Hat Ceph Storage の古いバージョンの一部は、Swift を使用する暗黙的なテナントのみをサポートします。

型

String

デフォルト

false

rgw_max_attr_name_len

説明

メタデータ名の最大長。0 はチェックをスキップします。

型

サイズ

デフォルト

0

rgw_max_attrs_num_in_req

説明

1 回のリクエストで入力できるメタデータ項目の最大数。

型

uint

デフォルト

0

rgw_max_attr_size

説明

メタデータ値の最大長。0 はチェックをスキップします

型

サイズ

デフォルト

0

rgw_swift_versioning_enabled

説明

Swift バージョニングを有効にします。

型

Boolean

デフォルト

0 または **1**

rgw_keystone_accepted_reader_roles

説明

読み取りのみに使用できるロールのリスト。

型

String

デフォルト

""

rgw_swift_enforce_content_length

説明

コンテナーをリストするときにコンテンツの長さを送信する

型

String

デフォルト

false`

A.8. LDAP 設定

名前	説明	型	例
rgw_ldap_uri	URI 形式の LDAP サーバーのスペース区切りリスト。	String	**ldaps://<ldap.your.domain>**
rgw_ldap_searchdn	ベースドメインとも呼ばれる LDAP 検索ドメイン名。	String	**cn=users,cn=accounts,dc=example,dc=com**
rgw_ldap_binddn	ゲートウェイはこの LDAP エントリー (ユーザー一致) でバインドします。	String	**uid=admin,cn=users,dc=example,dc=com**
rgw_ldap_secret	**rgw_ldap_binddn** の認証情報を含むファイル。	String	**/etc/openldap/secret**
rgw_ldap_dnattr	Ceph Object Gateway のユーザー名を含む LDAP 属性 (binddns 形式)。	String	**uid**