Red Hat Ceph Storage 7

对象网关指南

部署、配置和管理 Ceph 对象网关

Last Updated: 2024-07-02

Red Hat Ceph Storage 7 对象网关指南

部署、配置和管理 Ceph 对象网关

法律通告

摘要

本文档提供有关部署、配置和管理 Ceph 对象网关环境的指导。本指南使用"Day Zero（第 0 天）"、"Day One（第 1 天）"和"Day Two（第 2 天）"组织方法，为读者提供逻辑进度路径。第 0 天是在实施潜在解决方案之前进行的研究和规划的地方，请参见第 1章和第 2 章。第 1 天是实际部署和安装软件的时间，请参见第 3 章。第二天是所有基本配置和高级配置发生的地方，请参见第 4、5 和 6 章。 红帽致力于替换我们的代码、文档和 Web 属性中存在问题的语言。我们从这四个术语开始：master、slave、黑名单和白名单。由于此项工作十分艰巨，这些更改将在即将推出的几个发行版本中逐步实施。详情请查看 CTO Chris Wright 信息

目录

第 1 章 CEPH 对象网关

Ceph 对象网关（也称为 RADOS 网关(RGW)）是在 **librados** 库基础上构建的对象存储接口，为应用提供 Ceph 存储群集的 RESTful 网关。Ceph 对象网关支持三个接口：

S3 兼容性：

通过与 Amazon S3 RESTful API 的大子集兼容的接口提供对象存储功能。

您可以运行 S3 选择来加快吞吐量。用户可以在没有介质器的情况下直接运行 S3 选择查询。有两个 S3 选择工作流，一个用于 CSV，一个用于 Apache Parquet (Parquet)，它们为 CSV 和 Parquet 对象提供 S3 选择操作。有关这些 S3 选择操作的更多信息，请参阅 *Red Hat Ceph Storage Developer Guide* 中的 *S3 选择操作*。

Swift 兼容性：

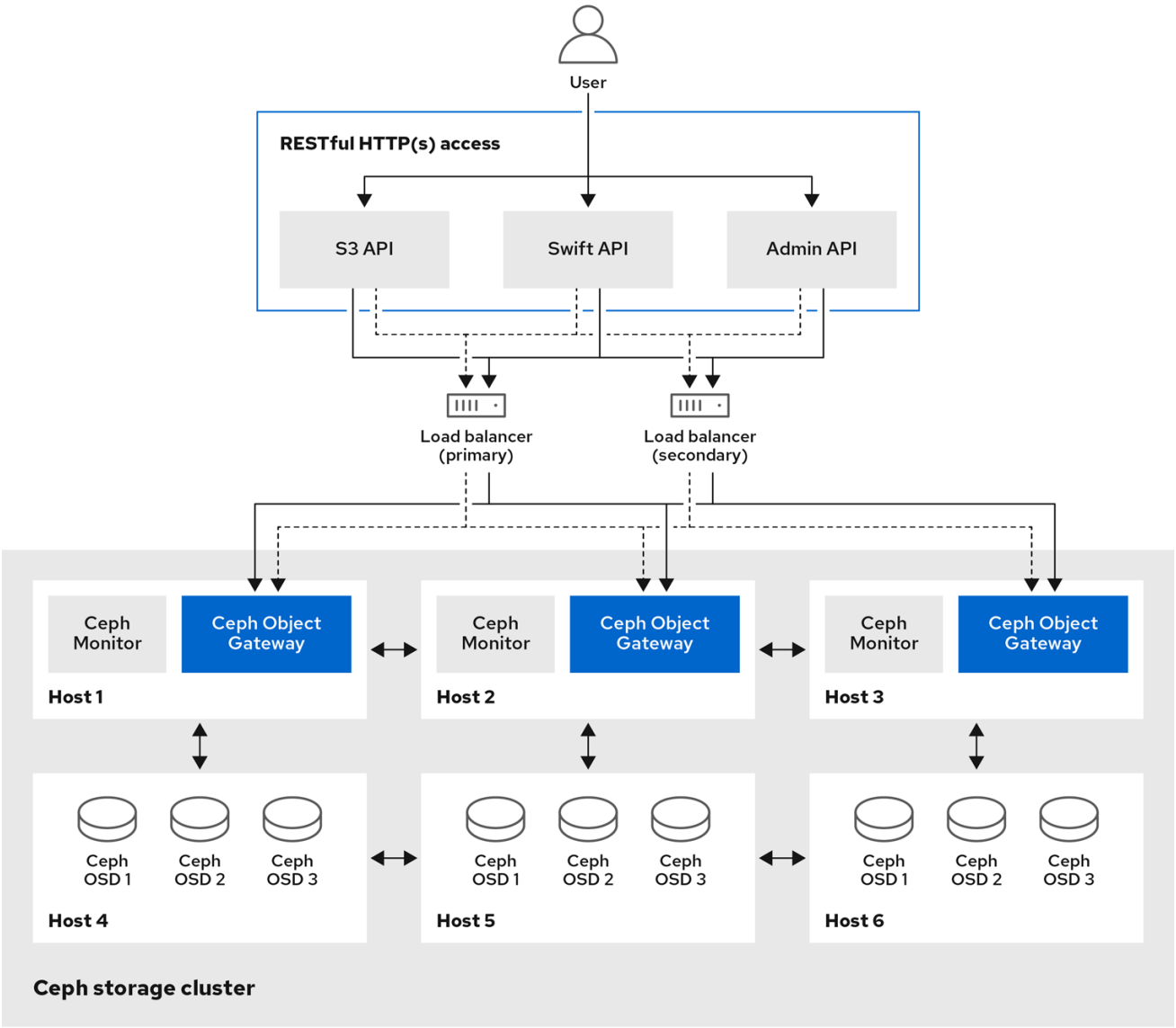
通过与 OpenStack Swift API 的大集兼容的接口提供对象存储功能。

Ceph 对象网关是一种与 Ceph 存储群集交互的服务。由于它提供与 OpenStack Swift 和 Amazon S3 兼容的接口，Ceph 对象网关拥有自己的用户管理系统。Ceph 对象网关可以将数据存储到同一个 Ceph 存储群集中，用于存储来自 Ceph 块设备客户端的数据；但是，它将涉及单独的池，可能会涉及到不同的 CRUSH 层次结构。S3 和 Swift API 共享一个通用命名空间，因此您可以使用一个 API 写入数据并与其他 API 检索数据。

管理 API:

提供用于管理 Ceph 对象网关的管理界面。

管理 API 请求在以 **admin** 资源端点开头的 URI 上执行。管理 API 的授权模仿 S3 授权约定。有些操作要求用户具有特殊的管理功能。响应类型可以是 XML 或 JSON，方法是在请求中指定 format 选项，但默认为 JSON 格式。



WORM 简介

write-Once-Read-Many (WORM)是一种安全数据存储模型，用于在生产区域中破坏对象和 bucket 时也用于保证数据保护和数据检索。

在 Red Hat Ceph Storage 中，数据安全性是通过使用带有只读功能的 S3 Object Lock 实现的，该功能用于使用 Write-Once-Read-Many (WORM)模型存储对象和 bucket，防止它们被删除或覆盖。即使 Red Hat Ceph Storage 管理员也不能删除它们。

S3 对象锁定提供两种保留模式：

- 监管
- COMPLIANCE

这些保留模式对您的对象应用不同的保护级别。您可以将保留模式应用到由 Object Lock 保护的任何对象版本。

在 GOVERNANCE 中，用户无法覆盖或删除对象版本，或者更改其锁定设置，除非他们具有特殊权限。使用 GOVERNANCE 模式，您可以防止大多数用户删除对象，但您仍可以授予一些用户权限来更改保留设置或删除对象（如果需要）。

在 COMPLIANCE 模式中，任何用户无法覆盖或删除受保护的对象版本。当对象被锁定为 COMPLIANCE 模式时，其保留模式无法更改或缩短。

其它资源

- 如需了解更多详细信息，请参阅 *Red Hat Ceph Storage 对象网关指南中的 为 S3 启用对象锁定*。

第 2 章 注意事项和建议

作为存储管理员，在运行 Ceph 对象网关和实施多站点 Ceph 对象网关解决方案之前需要考虑什么基本了解非常重要。您可以了解硬件和网络要求，了解哪种类型的工作负载与 Ceph 对象网关完美配合，以及红帽的建议。

先决条件

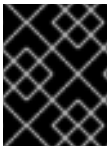
- 了解、考虑和规划存储解决方案的时间.

2.1. RED HAT CEPH STORAGE 的网络注意事项

云存储解决方案的一个重要方面是存储集群可能会因为网络延迟及其他因素而耗尽 IOPS。另外，存储集群可能会因为带宽限制而无法在存储集群用尽存储容量前耗尽吞吐量。这意味着网络硬件配置必须支持所选工作负载，以满足价格与性能要求。

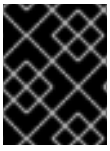
存储管理员希望存储集群尽快恢复。仔细考虑存储集群网络的带宽要求、通过订阅的网络链接，以及隔离客户端到集群流量的集群内部流量。在考虑使用 Solid State Disks(SSD)、闪存、NVMe 和其他高性能存储设备时，还需要考虑到网络性能变得越来越重要。

Ceph 支持公共网络和存储集群网络。公共网络处理客户端流量以及与 Ceph 监控器的通信。存储集群网络处理 Ceph OSD 心跳、复制、回填和恢复流量。**至少**，存储硬件应使用 10 GB 的以太网链接，您可以为连接和吞吐量添加额外的 10 GB 以太网链接。



重要

红帽建议为存储集群网络分配带宽，以便它是使用 **osd_pool_default_size** 作为复制池多个池基础的公共网络的倍数。红帽还建议在单独的网卡中运行公共和存储集群网络。



重要

红帽建议在生产环境中使用 10 GB 以太网部署 Red Hat Ceph Storage。1 GB 以太网网络不适用于生产环境的存储集群。

如果出现驱动器故障，在 1 GB 以太网网络中复制 1 TB 数据需要 3 小时，3 TB 需要 9 小时。使用 3 TB 是典型的驱动器配置。相比之下，使用 10 GB 以太网网络，复制时间分别为 20 分钟和 1 小时。请记住，当 Ceph OSD 出现故障时，存储集群通过将包含的数据复制到同一故障域中其他 OSD，以及作为故障 OSD 的设备类来恢复。

对于大型环境（如机架）的故障，意味着存储集群将使用的带宽要高得多。在构建由多个机架组成的存储群集（对于大型存储实施常见）时，应考虑在"树树"设计中的交换机之间利用尽可能多的网络带宽，以获得最佳性能。典型的 10 GB 以太网交换机有 48 10 GB 端口和四个 40 GB 端口。使用 40 GB 端口以获得最大吞吐量。或者，考虑将未使用的 10 GB 端口和 QSFP+ 和 SFP+ 电缆聚合到 40 GB 端口，以连接到其他机架和机械路由器。此外，还要考虑使用 LACP 模式 4 来绑定网络接口。另外，使用巨型帧、最大传输单元 (MTU) 9000，特别是在后端或集群网络上。

在安装和测试 Red Hat Ceph Storage 集群之前，请验证网络吞吐量。Ceph 中大多数与性能相关的问题通常是因为网络问题造成的。简单的网络问题（如粒度或 Bean Cat-6 电缆）可能会导致带宽下降。至少将 10 GB 以太网用于前端网络。对于大型集群，请考虑将 40 GB ethernet 用于后端或集群网络。

重要

为了优化网络，红帽建议使用巨型帧来获得更高的每带宽比率的 CPU，以及一个非阻塞的网络交换机后端。Red Hat Ceph Storage 在通信路径的所有网络设备中，公共和集群网络需要相同的 MTU 值。在在生产环境中使用 Red Hat Ceph Storage 集群之前，验证环境中所有主机和网络设备上的 MTU 值相同。

2.2. RED HAT CEPH STORAGE 的基本注意事项

使用 Red Hat Ceph Storage 的第一个考虑因素是为数据制定存储策略。存储策略是一种存储服务特定用例的数据的方法。如果您需要为 OpenStack 等云平台存储卷和镜像，可以选择将数据存储在带有 Solid State Drives (SSD) 的快速 Serial Attached SCSI (SAS) 驱动器上。相反，如果您需要存储 S3 或 Swift 兼容网关的对象数据，您可以选择使用更经济的方式，如传统的 SATA 驱动器。Red Hat Ceph Storage 可以在同一存储集群中同时容纳这两种场景，但您需要一种方式为云平台提供快速存储策略，并为对象存储提供更传统的存储方式。

一个成功的 Ceph 部署中的最重要的一个步骤是，找出一个适合存储集群的用例和工作负载的性价比配置集。为用例选择正确的硬件非常重要。例如，为冷存储应用程序选择 IOPS 优化的硬件会不必要地增加硬件成本。然而，在 IOPS 密集型工作负载中，选择容量优化的硬件使其更具吸引力的价格点可能会导致用户对性能较慢的抱怨。

Red Hat Ceph Storage 可以支持多种存储策略。用例、成本与好处性能权衡以及数据持久性是帮助开发合理存储策略的主要考虑因素。

使用案例

Ceph 提供大量存储容量，它支持许多用例，例如：

- Ceph 块设备客户端是云平台的领先存储后端，可为具有写时复制（copy-on-write）克隆等高性能功能的卷和镜像提供无限存储。
- Ceph 对象网关客户端是云平台的领先存储后端，为音频、位映射、视频和其他数据等对象提供 RESTful S3 兼容和 Swift 兼容对象存储。
- 传统文件存储的 Ceph 文件系统.

成本比较性能优势

越快越好。越大越好。越耐用越好。但是，每种出色的质量、相应的成本与收益权衡都有价格。从性能角度考虑以下用例：SSD 可以为相对较小的数据和日志量提供非常快速的存储。存储数据库或对象索引可以从非常快的 SSD 池中受益，但对于其他数据而言成本过高。带有 SSD 日志的 SAS 驱动器以经济的价格为卷和图像提供快速性能。没有 SSD 日志地 SATA 驱动器可提供低成本存储，同时整体性能也较低。在创建 OSD 的 CRUSH 层次结构时，您需要考虑用例和可接受的成本与性能权衡。

数据持续时间

在大型存储集群中，硬件故障是预期的，而非例外。但是，数据丢失和服务中断仍然不可接受。因此，数据的持久性非常重要。Ceph 通过对象的多个副本解决数据持久性问题，或使用纠删代码和多个编码区块来解决数据持久性。多个副本或多个编码区块会带来额外的成本与好处权衡：存储更少的副本或编码区块会更便宜，但可能会导致在降级状态中为写入请求提供服务。通常，一个具有两个额外副本的对象（或两个编码区块）可以允许存储集群在存储集群恢复时服务降级状态的写入。

在出现硬件故障时，复制存储在故障域中的一个或多个数据冗余副本。但是，冗余的数据副本规模可能会变得昂贵。例如，要存储 1 PB 字节并带有三倍复制的数据，将需要至少具有 3 PB 存储容量的集群。

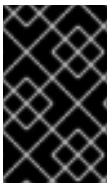
纠删代码将数据存储为数据区块和编码区块。如果数据区块丢失，纠删代码可以使用剩余的数据区块和编码区块来恢复丢失的数据区块。纠删代码比复制更经济。例如，使用带有 8 个数据区块和 3 个编码区块的

纠删代码提供与 3 个数据副本相同的冗余。但是，与复制相比（使用 3 倍的初始数据），此类编码方案使用约 1.5 倍的初始数据。

CRUSH 算法通过确保 Ceph 将额外的副本或编码区块存储在存储集群内的不同位置来协助这个过程。这样可确保单个存储设备或主机的故障不会丢失防止数据丢失所需的所有副本或编码区块。您可以规划一个成本取舍存储策略，以及数据持久性，然后将它作为存储池呈现给 Ceph 客户端。

重要

数据存储池可以使用纠删代码。存储服务数据和存储桶索引的池使用复制。



重要

与 Ceph 的对象复制或编码区块相比，RAID 解决方案已变得过时。不要使用 RAID，因为 Ceph 已经处理数据持久性，降级的 RAID 对性能有负面影响，并且使用 RAID 恢复数据比使用深度副本或纠删代码区块要慢得多。

其它资源

- 如需了解更多详细信息，请参阅 *Red Hat Ceph Storage 安装指南中的 Red Hat Ceph Storage 的最低硬件注意事项* 部分。

2.2.1. Colocating Ceph 守护进程及其优点

您可以在同一主机上并置容器化 Ceph 守护进程。以下是并置某些 Ceph 守护进程的优点：

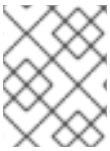
- 显著提高规模较小的总拥有成本(TCO)。
- 可以提高整体性能。
- 减少物理主机数量以实现最低配置。
- 更好的资源利用率。
- 升级 Red Hat Ceph Storage 更加简单。

通过使用容器，您可以将以下列表中的一个守护进程与 Ceph OSD 守护进程(**ceph-osd**)并置。此外，对于 Ceph 对象网关(**radosgw**)、Ceph 元数据服务器(**ceph-mds**)和 Grafana，您可以将它与 Ceph OSD 守护进程并置，以及以下列表中的守护进程。

- Ceph 元数据服务器 (**ceph-mds**)
- Ceph Monitor (**ceph-mon**)
- Ceph 管理器(**ceph-mgr**)
- NFS Ganesha (**nfs-ganesha**)
- Ceph 管理器(**ceph-grafana**)

表 2.1. 守护进程放置示例

主机名	Daemon	Daemon	Daemon
host1	OSD	监控和管理器	Prometheus
host2	OSD	监控和管理器	RGW
host3	OSD	监控和管理器	RGW
host4	OSD	Metadata Server	
host5	OSD	Metadata Server	



注意

由于 **ceph-mon** 和 **ceph-mgr** 协同工作，因此它们不会被视为两个独立的守护进程，以满足 colocation 的目的。

可以在命令行界面中使用 **--placement** 选项对 **ceph orch** 命令执行 Ceph 守护进程，也可以使用服务规格 YAML 文件。

命令行示例

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

服务规格 YAML 文件示例

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

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

红帽建议将 Ceph 对象网关与 Ceph OSD 容器共存以提高性能。要在不增加硬件成本的情况下获得最高的性能，请每个主机使用两个 Ceph 对象网关守护进程。

Ceph 对象网关命令行示例

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

Ceph 对象网关服务规格 YAML 文件示例

```
service_type: rgw
service_id: example
placement:
  count: 6
  hosts:
```

```
- host01
- host02
- host03
```

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

下图显示了具有并置守护进程和非并置守护进程的存储集群之间的区别。

图 2.1. colocated Daemons

图 2.2. 非并置守护进程

其他资源

- 有关使用 **--placement** 选项的更多详细信息，*请参阅 Red Hat Ceph Storage Operations 指南中的使用 Ceph Orchestrator 管理服务* 章节。
- 如需更多信息，请参阅 *Red Hat Ceph Storage RGW 部署策略和大小指南*。

2.3. RED HAT CEPH STORAGE 工作负载注意事项

Ceph 存储集群的一个关键优势在于能够使用性能域支持同一存储集群中的不同类型的工作负载。不同的硬件配置可以与每个性能域关联。存储管理员可以在适当的性能域中部署存储池，为应用提供专为特定性能和成本配置文件量身定制的存储。为这些性能域选择适当的大小和优化的服务器是设计 Red Hat Ceph Storage 集群的一个重要方面。

在读取和写入数据的 Ceph 客户端接口中，Ceph 存储集群显示为一个客户端存储数据的简单池。但是，

存储集群以对客户端接口完全透明的方式执行许多复杂的操作。Ceph 客户端和 Ceph 对象存储守护进程（称为 Ceph OSD）或只是 OSD，都使用可扩展哈希下的受控复制 (CRUSH) 算法来存储和检索对象。Ceph OSD 可以在存储集群内的容器中运行。

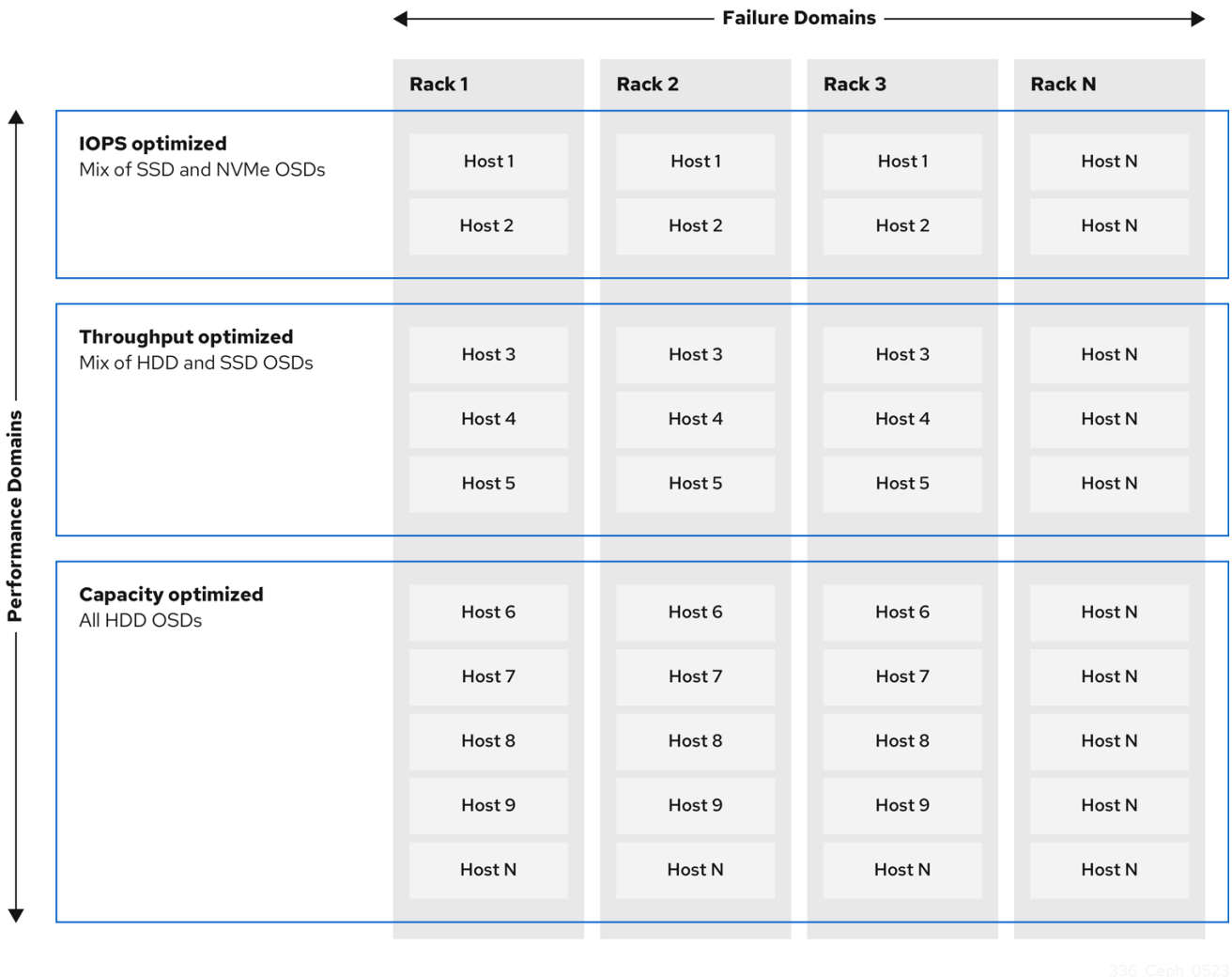
CRUSH map 描述了集群资源的拓扑结构，并且 map 存在于客户端主机和集群中的 Ceph 监控主机中。Ceph 客户端和 Ceph OSD 都使用 CRUSH map 和 CRUSH 算法。Ceph 客户端直接与 OSD 通信，消除了集中式对象查找和潜在的性能瓶颈。利用 CRUSH map 并与其对等方通信，OSD 可以处理复制、回填和恢复，从而实现动态故障恢复。

Ceph 使用 CRUSH map 来实施故障域。Ceph 还使用 CRUSH map 实施性能域，这只需将底层硬件的性能配置文件纳入考量。CRUSH map 描述了 Ceph 存储数据的方式，它作为简单的层次结构（特别是圆环图和规则集）实施。CRUSH map 可以支持多种层次结构，将一种类型的硬件性能配置集与另一类分隔开。Ceph 实施具有设备"类"的性能域。

例如，您可以让这些性能域共存在同一 Red Hat Ceph Storage 集群中：

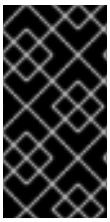
- 硬盘 (HDD) 通常适合以成本和容量为导向的工作负载。
- 吞吐量敏感的工作负载通常使用 HDD，在固态驱动器 (SSD) 上 Ceph 写入日志。
- MySQL 和 MariaDB 等 IOPS 密集型工作负载通常使用 SSD。

图 2.3. 性能和故障域



工作负载

Red Hat Ceph Storage 针对三种主要工作负载进行了优化：



重要

在购买硬件**前**，请仔细考虑由 Red Hat Ceph Storage 运行的工作负载，因为它可能会显著影响存储集群的价格和性能。例如，如果工作负载是容量优化的，并且硬件更适合通过吞吐量优化的工作负载，则硬件的成本将超过必要成本。相反，如果工作负载被优化吞吐量，且硬件更适合容量优化的工作负载，则存储集群的性能会受到影响。

- **优化 IOPS：** IOPS（Input, output per second）优化部署适合云计算操作，例如将 MYSQL 或 MariaDB 实例作为 OpenStack 上的虚拟机运行。优化 IOPS 部署需要更高的性能存储，如 15k RPM SAS 驱动器和单独的 SSD 日志，以处理频繁的写入操作。一些高 IOPS 情景使用所有闪存存储来提高 IOPS 和总吞吐量。
  IOPS 优化存储集群具有以下属性：
  - 每个 IOPS 的成本最低。
  - 每 GB 的 IOPS 最高。
  - 99 个百分点延迟一致性.

  IOPS 优化存储集群的用例：
  - 典型的块存储.
  - 用于硬盘 (HDD) 或 2x 复制的 3 倍复制，用于固态硬盘 (SSD)。
  - OpenStack 云上的 MySQL.
- **优化吞吐量：**使用优化吞吐量的部署适合服务大量数据，如图形、音频和视频内容。优化吞吐量的部署需要高带宽网络硬件、控制器和硬盘，具有快速顺序的读写特征。如果要求快速数据访问，则使用吞吐量优化存储策略。此外，如果要求快速写入性能，将 Solid State Disk (SSD) 用于日志将显著提高写入性能。
  吞吐量优化存储集群具有以下属性：
  - 每 MBps 成本最低（吞吐量）。
  - 每个 TB 的 MBps 最高。
  - 每个 BTU 的 MBps 最高。
  - 每个 Watt 的 MBps 最高。
  - 97% 的延迟一致性.

  优化吞吐量的存储集群用例：
  - 块或对象存储。
  - 3 倍复制。
  - 面向视频、音频和图像的主动性能存储.
  - 流媒体，如 4k 视频.

- **优化容量：** 容量优化部署适合以尽可能低的成本存储大量数据。容量优化的部署通常会以更具吸引力的价格点来换取性能。例如，容量优化部署通常使用速度较慢且成本更低的 SATA 驱动器和共同定位日志，而不是使用 SSD 进行日志。
  成本和容量优化的存储集群具有以下属性：

  - 每 TB 成本最低。
  - 每 TB 的 BTU 最低。
  - 每 TB 的 Watts 最低.

  用于成本和容量优化的存储集群有：

  - 典型的对象存储.
  - 纠删代码，以最大程度地提高可用容量
  - 对象存档。
  - 视频、音频和图像对象存储库.

2.4. CEPH 对象网关注意事项

设计存储集群的另一个重要方面是确定存储集群会位于一个数据中心站点，还是跨越多个数据中心站点。多站点存储群集得益于地理分散的故障切换和灾难恢复，如长期停电、农业、风暴、风暴、水水或其他灾难。另外，多站点存储群集可以具有主动配置，该配置可将客户端应用定向到最接近的可用存储集群。这是内容交付网络的良好存储策略。考虑尽可能将数据放置在客户端附近。这对于吞吐量密集型工作负载非常重要，如 4k 视频。

重要

红帽建议识别域、zone group 和 zone 名称 BEFORE 创建 Ceph 的存储池。使用区域名称作为标准命名约定附加一些池名称。

其它资源

- 如需更多信息，请参阅 *Red Hat Ceph Storage 对象网关 指南中的多站点配置和管理* 部分。

2.4.1. 管理数据存储

Ceph 对象网关将管理数据存储在实例的区域配置中定义的一系列池中。例如，后续小节中讨论的 bucket、用户、用户配额和使用量统计存储在 Ceph 存储集群的池中。默认情况下，Ceph 对象网关创建以下池，并将它们映射到默认区域。

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

- **.default.rgw.buckets.non-ec**

注意

只有 Ceph 对象网关中创建 bucket 之后，才会创建 **.default.rgw.buckets.index** 池，而数据上传到 bucket 后才会创建 **.default.rgw.buckets.data** 池。

考虑手动创建这些池，以便您可以设置 CRUSH 规则集和放置组的数量。在典型的配置中，存储 Ceph 对象网关管理数据的池通常使用相同的 CRUSH 规则集，并且使用较少的 PG，因为管理数据有 10 个池。

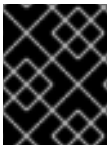
红帽建议 **.rgw.root** 池和服务池使用相同的 CRUSH 层次结构，并且至少将 **node** 用作 CRUSH 规则中的故障域。红帽建议将 **复制** 用于数据持久性，而不要**删除 .rgw.root** 池和服务池。

如果您将太多 PG 分配给池，**mon_pg_warn_max_per_osd** 设置将发出警告，默认为 **300**。您可以调整值以满足您的需求和硬件的功能，其中 **n** 是每个 OSD 的最大 PG 数。

```
mon_pg_warn_max_per_osd = n
```

注意

对于包括 **.rgw.root** 在内的服务池，*每个池计算器的 Ceph 放置组(PG)* 建议的 PG 数显著低于每个 Ceph OSD 的目标 PG。此外，还要确保计算器的第 4 步中设置 Ceph OSD 的数量。



重要

垃圾回收使用带有常规 RADOS 对象的 **.log** 池，而不是 OMAP。在未来的发行版中，更多功能会将元数据存储到 **.log** 池。因此，红帽建议将 NVMe/SSD Ceph OSD 用于 **.log** 池。

.rgw.root 池

存储 Ceph 对象网关配置的池。这包括 realm、zone group 和 zone。按照惯例，其名称不会以区域名称开头。

服务池

服务池存储与服务控制、垃圾收集、日志记录、用户信息和使用量相关的对象。按照惯例，这些池名称的前置为池名称的区域名称。

- ***.ZONE_NAME*.rgw.control** : 控制池。
- ***.ZONE_NAME*.log** ：日志池包含所有存储桶、容器和对象操作的日志，如 create、read、update 和 delete。
- ***.ZONE_NAME*.rgw.buckets.index** ：此池存储存储桶的索引。
- ***.ZONE_NAME*.rgw.buckets.data** ：此池存储存储桶的数据。
- ***.ZONE_NAME*.rgw.meta** ：元数据池存储 user_keys 和其他关键元数据。
- ***.ZONE_NAME*.meta:users.uid** ：用户 ID 池包含唯一用户 ID 的映射。
- ***.ZONE_NAME*.meta:users.keys** ：密钥池包含每个用户 ID 的访问密钥和密钥。
- ***.ZONE_NAME*.meta:users.email** ：电子邮件池包含与用户 ID 关联的电子邮件地址。

- ***.ZONE_NAME*.meta:users.swift** ：Swift 池包含用于用户 ID 的 Swift 子用户信息。

其它资源

- 如需了解更多详细信息，请参阅 *Red Hat Ceph Storage 对象网关指南中的 关于池* 部分。
- 如需了解更多详细信息，请参阅 *Red Hat Ceph Storage 策略指南*。

2.4.2. 索引池

当选择用于 Ceph 对象网关的 OSD 硬件时，无论用例 - 至少需要一个高性能驱动器（可以是 SSD 或 NVMe 驱动器）的 OSD 节点来存储索引池。当 bucket 包含大量对象时，这一点尤为重要。

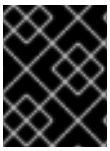
对于运行 Bluestore 的 Red Hat Ceph Storage，红帽建议将 NVMe 驱动器部署为 **block.db** 设备，而不是作为单独的池。

Ceph 对象网关索引数据仅写入到对象映射(OMAP)中。BlueStore 的 OMAP 数据驻留在 OSD 上的 **block.db** 设备上。当 NVMe 驱动器作为 HDD OSD 的 **block.db** 设备以及索引池由 HDD OSD 支持时，索引数据将仅写入 **block.db** 设备。只要 **block.db** 分区/lvm 的大小被正确设置为 4% 的块，则这个配置是 BlueStore 唯一需要的。

注意

红帽不支持索引池的 HDD 设备。如需有关支持的配置的更多信息，请参阅 *Red Hat Ceph Storage: 支持的配置* 文章。

索引条目大约是 200 字节的数据，存储为 **rocksdb** 中的 OMAP。虽然这是微不足的数据，但有些使用 Ceph 对象网关可能会导致单个 bucket 中有几十或几百个对象。通过将索引池映射到具有高性能存储介质的 CRUSH 层次结构，当 bucket 包含大量对象时，对则延的减小会显著提高性能。



重要

在生产环境中，典型的 OSD 节点将至少有一个 SSD 或 NVMe 驱动器来存储 OSD 日志，以及索引池或 **block.db** 设备，其对同一物理驱动器使用单独的分区或逻辑卷。

2.4.3. 数据池

数据池是 Ceph 对象网关存储特定存储策略的对象数据的位置。数据池具有完整放置组 (PG)，而不是用于服务池的数量减少的 PG。考虑将纠删代码用于数据池，因为它比复制更高效，而且可以显著降低容量要求，同时保持数据持久性。

要使用纠删代码，请创建一个纠删代码 profile。如需了解更多详细信息，请参阅 *Red Hat Ceph Storage 策略指南中的 Erasure Code Profiles* 部分。

重要

选择正确的配置集非常重要，因为您创建池后无法更改配置集。若要修改配置文件，您必须创建一个具有不同配置文件的新池，并将对象从旧池中迁移到新池中。

默认配置是两个数据区块和一个编码区块，这意味着只能丢失一个 OSD。对于更高的弹性，请考虑大量数据和编码区块。例如，一些大型系统使用 8 个数据区块和 3 个编码区块，这允许 3 个 OSD 在不丢失数据的情况下失败。

重要

每个数据和编码区块 SHOULD 至少会存储在不同的节点或主机上。对于较小的存储集群，这会使使用 **机架** 作为更多数据和编码区块的最小 CRUSH 故障域不切实际。因此，数据池通常使用单独的 CRUSH 层次结构并将 **主机** 用作最小 CRUSH 故障域。红帽建议 **host** 作为最小故障域。如果纠删代码区块存储在同一主机上的 Ceph OSD 上，主机故障（如失败的日志或网卡）可能会导致数据丢失。

若要创建数据池，运行 **ceph osd pool create** 命令，使用池名称、PG 和 PGP 的数量、 **erasure** 数据持久性方法、纠删代码 profile 和规则名称。

2.4.4. 数据额外池

data_extra_pool 用于无法使用纠删代码的数据。例如，多部分上传允许上传大型对象，如多个部分的 movie。这些部分必须首先存储而无纠删代码。纠删代码应用到整个对象，而不是部分上传。

注意

每个池的 PG 数量约为 **data_extra_pool** 的每个池的 PG 数量(PG)；但 PG 计数大约是服务池的 PG 数量两倍，与 bucket 索引池相同。

若要创建数据额外池，请使用池名称、PG 和 PGP 的数量、**复制** 数据持久性方法以及规则名称运行 **ceph osd pool create** 命令。例如：

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

2.5. 开发 CRUSH 层次结构

作为存储管理员，在部署 Ceph 存储集群和对象网关时，Ceph 对象网关通常具有默认的 zone group 和 zone。Ceph 存储群集将具有默认的池，后者使用 CRUSH 层次结构和默认 CRUSH 规则的 CRUSH map。

重要

默认 **rbd** 池可以使用默认的 CRUSH 规则。如果 Ceph 客户端已使用它们存储客户端数据，请不要 **删除** 默认规则或层次结构。

生产网关通常使用自定义域、zone group 和 zone，具体取决于网关的使用和地理位置。此外，Ceph 存储群集将具有具有多个 CRUSH 层次结构的 CRUSH map。

- **服务池：** 至少一个 CRUSH 层次结构将用于服务池，并且可能用于数据。服务池包含 **.rgw.root** 以及与区域关联的服务池。服务池通常位于单个 CRUSH 层次结构下，并使用复制来实现数据持久性。数据池也可能使用 CRUSH 层次结构，但池通常配置有纠删代码以实现数据持久性。

- **索引：** 至少有一个 CRUSH 层次结构 **SHOULD** 用于索引池，其中 CRUSH 层次结构映射到高性能介质，如 SSD 或 NVMe 驱动器。bucket 索引可能会成为性能瓶颈。红帽建议在 CRUSH 层次结构中使用 SSD 或 NVMe 驱动器。在用于 Ceph OSD 日志的 SSD 或 NVMe 驱动器上创建索引分区。另外，索引应该配置有存储桶分片。

- **放置池：** 每个放置目标的放置池包括 bucket 索引、数据存储桶和 bucket 额外内容。这些池可以属于单独的 CRUSH 层次结构。由于 Ceph 对象网关可以支持多种存储策略，存储策略的 bucket 池可能与不同的 CRUSH 层次结构关联，反映不同的用例，如 IOPS 优化、吞吐量优化和容量优

化。bucket 索引池 **SHOULD** 使用自己的 CRUSH 层次结构将 bucket 索引池映射到更高性能存储介质，如 SSD 或 NVMe 驱动器。

2.5.1. 创建 CRUSH roots

从管理节点上的命令行，为各个 CRUSH 层次结构在 CRUSH map 中创建 CRUSH roots。**必须** 至少有一个 CRUSH 层次结构，用于可能也提供数据存储池的服务池。**SHOULD** 至少有一个 CRUSH 层次结构用于 bucket 索引池，映射到高性能存储介质，如 SSD 或 NVMe 驱动器。

有关 CRUSH 层次结构的详细信息，请参见 *Red Hat Ceph Storage 策略指南 7* 中的 CRUSH *层次结构* 章节。

要手动编辑 CRUSH map，请参阅 *Red Hat Ceph Storage 策略指南 7* 中的 *编辑* CRUSH map 部分。

在以下示例中，名为 **data0**、**data1** 和 **data2** 的主机使用扩展逻辑名称，如 **data0-sas-ssd**、**data0-index** 等，因为 CRUSH 层次结构有多种指向同一物理主机的 CRUSH 层次结构。

典型的 CRUSH root 可能代表具有 SAS 驱动器和 SSD 的节点（用于日志）。例如：

```
##
# 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
}
```

用于 bucket 的 CRUSH root 索引 **SHOULD** 代表高性能介质，如 SSD 或 NVMe 驱动器。考虑在存储 OSD 日志的 SSD 或 NVMe 介质上创建分区。例如：

```
##
# 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. 在 CRUSH map 中使用逻辑主机名

在 RHCS 3 及更高版本中，CRUSH 支持存储设备"类"的概念，此概念在 RHCS 2 及早期版本中不受支持。在 RHCS 3 集群中，包含多类存储设备的主机或节点，如 NVMe、SSD 或 HDD，使用单一 CRUSH 层次结构和设备类别来区分不同类型的存储设备。这无需使用逻辑主机名。在 RHCS 2 和更早的版本中，

使用多个 CRUSH 层次结构，分别对应于每类设备，以及逻辑主机名，以区分 CRUSH 层次结构中的主机或节点。

在 RHCS 3 及更高版本中，CRUSH 支持存储设备"类"的概念，这在 RHCS 2 及更早的版本中不被支持。在 RHCS 3 集群中，主机或节点包含多个存储设备类，如 NVMe、SSD 或 HDD，使用单一 CRUSH 层次结构和设备类来区分不同的存储设备类别。这无需使用逻辑主机名。在 RHCS 2 和更早的版本中，使用多个 CRUSH 层次结构，分别对应于每类设备，以及逻辑主机名，以区分 CRUSH 层次结构中的主机或节点。

在 CRUSH map 中，主机名必须是唯一的，并且仅使用一次。当主机服务多个 CRUSH 层次结构和使用案例时，CRUSH map 可以使用逻辑主机名而不是实际的主机名，以确保主机名仅使用一次。例如，节点可能有多个类型的驱动器，如 SSD、SAS 驱动器和 SSD 日志，以及带有并置日志的 SATA 驱动器。要在 RHCS 2 及更早的版本中为同一主机创建多个 CRUSH 层次结构，层次结构需要使用逻辑主机名来代替实际主机名，使得存储桶名称在 CRUSH 层次结构中是唯一的。例如，如果主机名是 **data2**，CRUSH 层次结构可能会使用逻辑名称，如 **data2-sas-ssd** 和 **data2-index** ：

```
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
}
```

在示例中，主机 **data2** 使用逻辑名称 **data2-sas-ssd** 将带有 SSD 上日志的 SAS 驱动器映射到一个层次结构中。OSD ID **osd.0** 到 **osd.3** 代表在高吞吐量硬件配置中使用 SSD 日志的 SAS 驱动器。以下示例中的 OSD ID 与 OSD ID 不同。

在以下示例中，主机 **data2** 使用逻辑名称 **data2-index** 将 bucket 索引的 SSD 驱动器映射到第二个层次结构。OSD ID **osd.4** 代表 SSD 驱动器或其他高速度存储介质，专门用于 bucket 索引池。

```
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

在使用逻辑主机名时，请确保 Ceph 配置文件中存在以下设置之一，以防止 OSD 启动脚本在启动时使用实际主机名，因此无法在 CRUSH map 中查找数据。

当 CRUSH map 使用逻辑主机名时，如示例中所示，OSD 启动脚本会阻止 OSD 启动脚本在初始化时根据其实际主机名识别主机。在 Ceph 配置文件的 **[global]** 部分中，添加以下设置：

```
osd_crush_update_on_start = false
```

另一种定义逻辑主机名的方法是在 Ceph 配置文件的 **[osd.<ID>]** 部分中为各个 OSD 定义 CRUSH map 的位置。这将覆盖 OSD 启动脚本定义的任何位置。在示例中，条目可能类似如下：

```
[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

如果 CRUSH map 在重启时使用逻辑主机名而不是实际主机名时，Ceph Storage Cluster 假定 OSD 映射到实际主机名，则 CRUSH 映射中没有找到实际的主机名，Ceph 存储集群客户端将找不到 OSD 及其数据。

2.5.3. 创建 CRUSH 规则

与默认的 CRUSH 层次结构一样，CRUSH map 也包含默认的 CRUSH 规则。

注意

默认 **rbd** 池可能使用此规则。如果其他池已使用它存储客户数据，请不要删除默认规则。

有关 CRUSH 规则的一般详细信息，请参见 *Red Hat Ceph Storage 7 策略指南中的 CRUSH* 规则 部分。要手动编辑 CRUSH map，请参阅 Red Hat Ceph Storage 7 的 *Red Hat Ceph Storage 策略指南中的* 编辑 CRUSH map 部分。

对于每一 CRUSH 层次结构，创建一个 CRUSH 规则。下例演示了 CRUSH 层次结构的规则，该层次结构将存储服务池，包括 **.rgw.root**。在本例中，根 **sas-ssd** 充当 CRUSH 主层次结构。它使用名称 **rgw-service** 来区分其自身和默认规则。**step take sas-ssd** 行告知池使用 Creating CRUSH roots 中创建的 **sas-ssd** root，其子 bucket 包含具有 SAS 驱动器的 OSD 和高性能存储介质，如 SSD 或 NVMe 驱动器，用于高吞吐量硬件配置中的日志。**step chooseleaf** 的 **type rack** 部分是故障域。在以下示例中，这是一个机架。

```
##
# 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
}
```

注意

在示例中，如果数据被复制三次，集群中的至少应该有三个机架，其中包含相似数量的 OSD 节点。

提示

type replicated 设置与数据持久性、副本数或纠删代码 **无关**。仅支持 **replicated**。

下例演示了将存储数据池的 CRUSH 层次结构的规则。在本例中，根 **sas-ssd** 充当 CRUSH 主层次结构-与服务规则相同的 CRUSH 层次结构。它使用 **rgw-throughput** 来区分其自身与默认规则和 **rgw-service**。**step take sas-ssd** 行告知池使用 Creating CRUSH roots 中创建的 **sas-ssd** root，其子 bucket 包含具有 SAS 驱动器的 OSD 和高性能存储介质，如 SSD 或 NVMe 驱动器。**step chooseleaf** 的 **type host** 部分是故障域。在以下示例中，这是主机。注意该规则使用相同的 CRUSH 层次结构，但使用了不同的故障realm。

```
##
# 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
}
```

注意

在示例中，如果池将纠删代码与更多的数据进行纠删代码，且编码区块数超过默认值，则集群中的机架应至少包含数量相似的 OSD 节点，以便于纠删代码区块。对于较小的集群，这可能不实际，因此，示例中使用 **host** 作为 CRUSH 故障realm。

下例演示了 CRUSH 层次结构的规则，该规则将存储索引池。在本例中，根 **index** 充当 CRUSH 主层次结构。它使用 **rgw-index** 将自身与 **rgw-service** 和 **rgw-throughput** 区分开来。**step take index** 行告知池使用 Creating CRUSH Roots 创建 **index** root，其子存储桶包含高性能存储介质，如 SSD 或 NVMe 驱动器或 SSD 上存储 OSD 日志的 NVMe 驱动器或 NVMe 驱动器上。**step chooseleaf** 的 **type rack** 部分是故障域。在以下示例中，这是一个机架。

```
##
# 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
}
```

其它资源

- 有关 CRUSH 层次结构的一般详细信息，请参见 *Red Hat Ceph Storage 策略指南中的 CRUSH 管理* 章节。

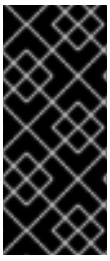
2.6. CEPH 对象网关多站点注意事项

Ceph 对象网关多站点配置至少需要两个 Red Hat Ceph Storage 集群，以及至少两个 Ceph 对象网关实例，每个 Red Hat Ceph Storage 集群都有一个。通常，两个 Red Hat Ceph Storage 集群将在地理上独立的位置；但是，相同的多站点配置可以在位于同一物理站点的两个 Red Hat Ceph Storage 集群中工作。

多站点配置需要一个主要 zone group 和一个主要 zone。另外，每个 zone group 都需要一个主 zone。zone group 可能具有一个或多个次要区域。

注意

您可以通过 CLI 或 Red Hat Ceph Storage 仪表板配置多站点。如需了解更多详细信息，*请参阅在 Ceph 仪表板上* 配置多站点对象网关。



重要

域的主要 zone group 中的主要区域负责存储域元数据的主副本，包括用户、配额和 bucket。此元数据会自动同步到 second zone 和 second zone group。通过 **radosgw-admin** 命令行界面(CLI) 执行的元数据操作**必须**在 primary zone 组的主区域的节点中执行，以确保它们与 second zone group 和 zone 同步。目前，*可以*对 second zone 和 zone group 分配元数据操作，但**不建议**这么做，因为它们**无法**同步，这可能导致元数据碎片。

下图说明了在多站点 Ceph 对象网关环境中可能采用的一种和两个域配置。

图 2.4. 一个域

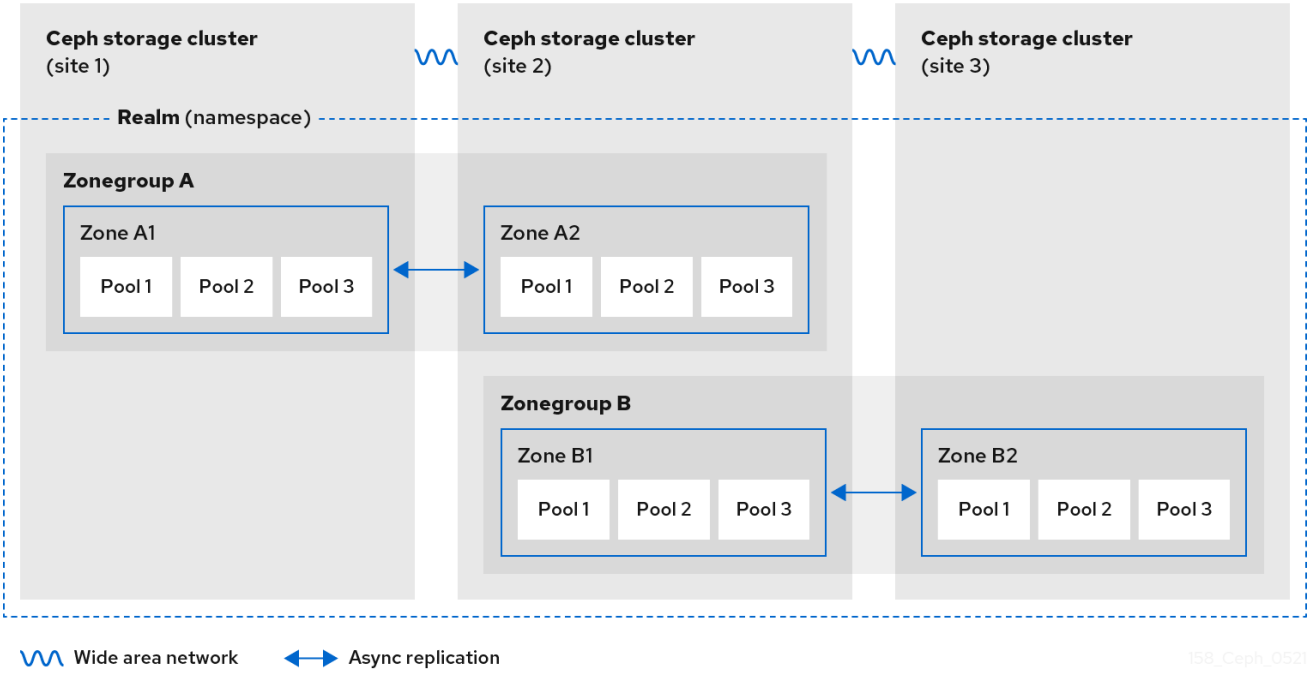


图 2.5. 两个 Realms

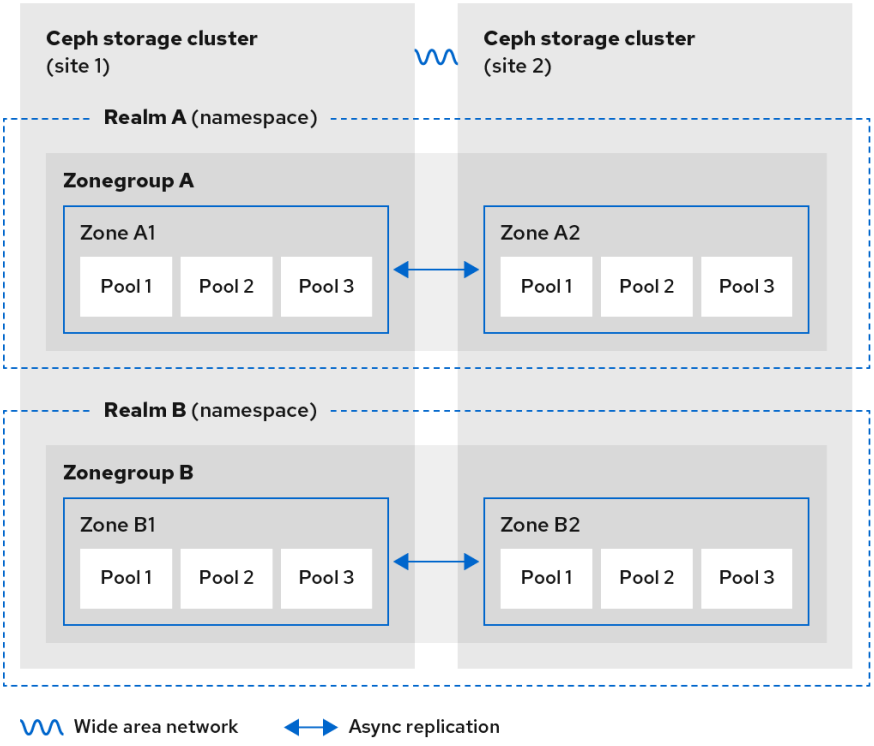
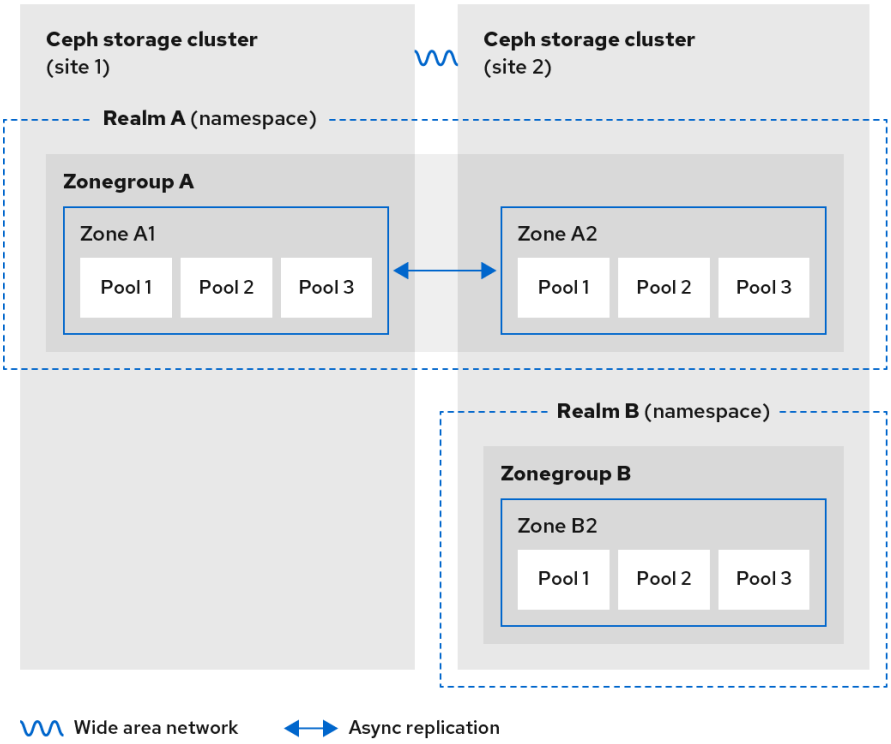


图 2.6. 两个 Realms Variant



2.7. 考虑存储大小

设计集群时最重要的因素之一是确定存储要求（大小）。Ceph 存储设计为可以扩展到 PB 以上。以下示例是 Ceph 存储集群的常见大小。

- **Small**: 250 TB

- **Medium**：1 Ptabyte
- **Large:** 2 PB 或更高

规模调整包括当前需求和近期的需求。请考虑网关客户端向群集添加新数据的速率。这可能因用例而异。例如，录制 4k 视频或存储医疗图像与较低存储密集型信息（如金融市场数据）相比，会更快地添加大量数据。另外，请考虑数据持久性方法（如复制与纠删代码）可能会对所需的存储介质产生重大影响。

有关大小调整的更多信息，请参阅 *Red Hat Ceph Storage 硬件指南* 及其用于选择 OSD 硬件的相关链接。

2.8. 考虑存储密度

Ceph 设计的另一个重要方面，包括存储密度。通常，存储集群在至少 10 个节点间存储数据，以确保在复制、回填和恢复时具有合理的性能。如果节点出现故障，存储集群中至少有 10 个节点，则只有 10% 的数据必须移动到存活的节点。如果节点数量大大减少，则必须将更高的数据百分比移到存活的节点。此外，还需要设置 **full_ratio** 和 **near_full_ratio** 选项，以适应节点故障，以确保存储集群可以写入数据。因此，务必要考虑存储密度。更高的存储密度不一定是一个不错的想法。

与更高的存储密度相比，另一个因素是纠删代码。当使用纠删代码编写对象并将 **节点** 用作最低 CRUSH 故障域时，Ceph 存储群集将需要尽可能多的节点，作为数据和编码区块。例如，使用 **k=8, m=3** 的集群应该至少有 11 个节点，每个数据或编码区块都存储在单独的节点上。

热插拔也是重要的考虑因素。大多数现代服务器都支持驱动器热插拔。但是，一些硬件配置需要删除多个驱动器来替换驱动器。红帽建议避免此类配置，因为它们可以在交换失败的磁盘时导致超过所需数量的 Ceph OSD。

2.9. 为 CEPH 监控节点考虑磁盘

Ceph 监控器使用 **rocksdb**，这对同步写入延迟非常敏感。红帽强烈建议使用 SSD 磁盘来存储 Ceph 监控数据。选择具有足够连续写入和吞吐量特征的 SSD 磁盘。

2.10. 调整回填和恢复设置

I/O 受到回填和恢复操作的负面影响，导致性能低下和最终用户不满意。要帮助满足集群扩展或恢复期间的 I/O 需求，请在 Ceph 配置文件中设置以下选项和值：

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

2.11. 调整集群映射大小

默认情况下，**ceph-osd** 守护进程缓存以前的 500 个 osdmaps。即使使用 deduplication，映射可能会每个守护进程消耗大量内存。在 Ceph 配置中调优缓存大小有助于显著减少内存消耗。例如：

```
[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
```

对于 Red Hat Ceph Storage 版本 3 及更高版本，**ceph-manager** 守护进程处理 PG 查询，因此 cluster map 不会影响性能。

2.12. 调整清理

默认情况下，Ceph 每周执行轻型清理和深度清理。轻型清理检查对象大小和校验和，以确保 PG 存储相同的对象数据。随着时间的推移，磁盘扇区的对象大小和校验和可能会变得不良。深度清理检查对象的副本内容，以确保实际内容相同。为此，深度清理以 **fsck** 的方式确保数据完整性，但该过程会对集群施加 I/O 罚款。甚至轻型清理也会影响 I/O。

默认设置可能允许 Ceph OSD 在规定时间（如峰值操作时间或负载过重的期间）启动清理。当清理操作与最终用户操作冲突时，最终用户可能会遇到延迟和性能不佳的情况。

为了防止最终用户性能下降，Ceph 提供了多个清理设置，可将清理限制为负载较低或非高峰时段的期间。详情请参阅 *Red Hat Ceph Storage 配置指南中的清理 OSD 部分*。

如果集群在一天中出现高负载，当晚晚出现低负载时，请考虑将清理限制为夜间小时。例如：

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

如果时间限制不是确定清理计划的有效方法，请考虑使用 **osd_scrub_load_threshold**。默认值为 **0.5**，但可以根据低负载状况进行修改。例如：

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

2.13. 增加 OBJECTER_INFLIGHT_OPS

为提高可扩展性，您可以编辑 **objecter_inflight_ops** 参数的值，该参数指定了允许的 I/O 请求的最大数量。这个参数用于客户端流量控制。

```
objecter_inflight_ops = 24576
```

2.14. 提高 RGW_THREAD_POOL_SIZE

为提高可扩展性，您可以编辑 **rgw_thread_pool_size** 参数的值，这是线程池的大小。新的 **beast** frontend 不受到线程池大小的限制，以接受新的连接。

```
rgw_thread_pool_size = 512
```

2.15. 在运行 CEPH 时调整 LINUX 内核的注意事项

生产环境的 Red Hat Ceph Storage 集群通常受益于操作系统调优，尤其是关于限值和内存分配。确保为存储集群中的所有主机进行了调整。您还可以在红帽支持下创建一个问题单，寻求其他指导。

增加文件描述符数量

如果 Ceph 对象网关缺少文件描述符，它可能会挂起。您可以修改 Ceph 对象网关主机上的 **/etc/security/limits.conf** 文件，以增加 Ceph 对象网关的文件描述符。

```
ceph      soft    nofile     unlimited
```

调整大型存储集群的 ulimit 值

在大型存储集群上运行 Ceph 管理命令时， 例如，带有 1024 个 Ceph OSD 或更多 OSD， 在每个运行管理命令的主机上创建一个 **/etc/security/limits.d/50-ceph.conf** 文件，其中包含以下内容：

```
USER_NAME      soft    nproc     unlimited
```

将 *USER_NAME* 替换为运行 Ceph 管理命令的非 root 用户帐户的名称。

注意

在 Red Hat Enterprise Linux 中，root 用户的 **ulimit** 值默认设置为 **ulimit**。

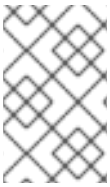
其它资源

- 有关 Ceph 的各种内部组件和这些组件的相关策略的详细信息，请参阅 *Red Hat Ceph Storage 策略指南*。

第 3 章 DEPLOYMENT

作为存储管理员，您可以使用 Ceph 编排器和命令行界面或服务规格部署 Ceph 对象网关。您还可以配置多站点 Ceph 对象网关，并使用 Ceph 编排器移除 Ceph 对象网关。

cephadm 命令将 Ceph 对象网关部署为守护进程集合，这些守护进程在多站点部署中管理单集群部署或特定的域和区域。



注意

使用 **cephadm** 时，Ceph 对象网关守护进程配置为使用 Ceph monitor 配置数据库，而不是 **ceph.conf** 文件或命令行选项。如果配置不在 **client.rgw** 部分中，则 Ceph 对象网关守护进程以默认设置启动并绑定到端口 **80**。

本节涵盖了以下管理任务：

- *使用命令行界面部署 Ceph 对象网关。*
- *使用服务规格部署 Ceph 对象网关。*
- *利用服务规范部署 Ceph 对象网关.*
- *使用 Ceph 编排器部署多站点 Ceph 对象网关。*
- *使用 Ceph 编排器移除 Ceph 对象网关.*
- *使用 Ceph 管理器 **rgw** 模块.*

先决条件

- 一个运行良好、健康的 Red Hat Ceph Storage 集群。
- 所有节点的根级别访问权限。
- 存储集群上的可用节点。
- 所有管理器、监视器和 OSD 都部署在存储集群中。

3.1. 使用命令行界面部署 CEPH 对象网关

利用 Ceph 编排器，您可以在命令行界面中使用 **ceph orch** 命令部署 Ceph 对象网关。

先决条件

- 一个正在运行的 Red Hat Ceph Storage 集群。
- 所有节点的根级别访问权限。
- 主机添加到集群中。
- 部署所有管理器、监控器和 OSD 守护进程。

流程

1. 登录到 Cephadm shell：

示例

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

2. 您可以通过三种不同的方式部署 Ceph 对象网关守护进程：

方法 1

- 创建 realm、zone group 和 zone，然后将放置规格与主机名搭配使用：

a. 创建一个域：

语法

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

示例

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

b. 创建区组：

语法

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

示例

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

c. 创建区：

语法

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

示例

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

d. 提交更改：

语法

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

示例

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

e. 运行 **ceph orch apply** 命令：

语法

```
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]"
```

示例

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

方法 2

- 使用任意服务名称为单个集群部署部署两个 Ceph 对象网关守护进程：

语法

```
ceph orch apply rgw SERVICE_NAME
```

示例

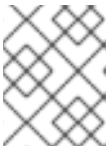
```
[ceph: root@host01 /]# ceph orch apply rgw foo
```

方法 3

- 在标记的一组主机上使用任意服务名称：

语法

```
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
```



注意

NUMBER_OF_DAEMONS 控制每个主机上部署的 Ceph 对象网关数量。要在不增加成本的情况下获得最高的性能，请将此值设置为 2。

示例

```
[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
```

验证

- 列出服务：

  示例

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

- 列出主机、守护进程和进程：

  语法

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

  示例

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

3.2. 使用 CEPH OBJECT STORAGE 后端部署 NFS 服务

您可以使用 Ceph 编排器在 Ceph 对象网关中部署 NFS 服务。

先决条件

- 一个正在运行的 Red Hat Ceph Storage 集群。
- 对 bootstrap 启动主机的 root 级别访问权限。
- 主机添加到集群中。
- 部署所有管理器、监控、Ceph 对象网关和 OSD 守护进程。

流程

1. 登录到 Cephadm shell：

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

2. 创建包含相关数据的 NFS 规格文件，包括需要安装 NFS 服务的主机：

   示例

   ```
   [root@host01 ~]# cat nfs-conf.yml

   service_type: nfs
   service_id: nfs-rgw-service
   placement:
     hosts: ['host1']
   spec:
     port: 2049
   ```

3. 通过在第 2 步中创建的规格文件应用 NFS 服务：

示例

```
[root@host01 ~]# ceph orch apply -i nfs-conf.yml
```

4. 验证 NFS 服务是否已成功创建：

示例

```
[root@host01 ~]# ceph orch ls --service_name nfs.nfs-rgw-service --service_type nfs
```

3.3. 使用服务规格部署 CEPH 对象网关

您可以使用服务规格和默认域、区域和区域组来部署 Ceph 对象网关。

先决条件

- 一个正在运行的 Red Hat Ceph Storage 集群。
- 对 bootstrap 启动主机的 root 级别访问权限。
- 主机添加到集群中。
- 部署所有管理器、监控和 OSD 守护进程。

流程

1. 作为 root 用户，创建规格文件：

示例

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

2. 编辑 **radosgw.yml** 文件，使其包含 default realm、zone 和 zone group 的以下详情：

语法

```
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
```

注意

NUMBER_OF_DAEMONS 控制每个主机上部署的 Ceph 对象网关数量。要在不增加成本的情况下获得最高的性能，请将此值设置为 2。

示例

```
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. 可选：对于自定义 realm、zone 和 zone group，请创建资源，然后创建 **radosgw.yml** 文件：

   a. 创建自定义 realm、zone 和 zone group：

      示例

      ```
      [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. 使用以下详细信息，创建 **radosgw.yml** 文件：

      示例

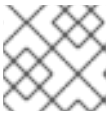
      ```
      service_type: rgw
      service_id: test_realm.test_zone
      placement:
        hosts:
        - host01
        - host02
        - host03
        count_per_host: 2
      spec:
        rgw_realm: test_realm
        rgw_zone: test_zone
        rgw_zonegroup: test_zonegroup
        rgw_frontend_port: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

-

4. 将 **radosgw.yml** 文件挂载到容器中的某个目录下：

   示例

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

   注意

   每次退出 shell 时，您都必须在部署守护进程前将该文件挂载到容器中。

5. 使用服务规格部署 Ceph 对象网关：

   语法

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

   示例

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

验证

- 列出服务：

  示例

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

- 列出主机、守护进程和进程：

  语法

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

  示例

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

3.4. 使用 CEPH 编排器部署多站点 CEPH 对象网关

Ceph 编排器支持 Ceph 对象网关的多站点配置选项。

您可以将每个对象网关配置为在主动区域配置中工作，从而允许写入到非主要区域。多站点配置存储在名为 realm 的容器中。

realm 存储 zone group、区域和一个时间周期。**rgw** 守护进程处理同步消除了对独立同步代理的需求，因此使用主动-主动配置运行。

您还可以使用命令行界面(CLI)部署多站点区域。

注意

以下配置假定在地理上至少有两个 Red Hat Ceph Storage 集群。但是，配置也在同一站点工作。

先决条件

- 至少两个正在运行的 Red Hat Ceph Storage 集群。
- 至少两个 Ceph 对象网关实例，每个实例对应一个 Red Hat Ceph Storage 集群。
- 所有节点的根级别访问权限。
- 节点或容器添加到存储集群中。
- 部署所有 Ceph 管理器、监控器和 OSD 守护进程。

流程

1. 在 **cephadm** shell 中，配置主区：
   a. 创建一个域：

      语法

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

      示例

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

      如果存储集群只有一个域，则指定 **--default** 标志。

   b. 创建主要区组：

      语法

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

      示例

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

   c. 创建一个主要区：

      语法

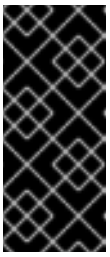
      ```
      radosgw-admin zone create --rgw-zonegroup=PRIMARY_ZONE_GROUP_NAME --rgw-
      zone=PRIMARY_ZONE_NAME --
      endpoints=http://RGW_PRIMARY_HOSTNAME:RGW_PRIMARY_PORT_NUMBER_1 --
      access-key=SYSTEM_ACCESS_KEY --secret=SYSTEM_SECRET_KEY
      ```

-

示例

```
[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. 可选：删除默认 zone、zone group 和关联的池。



重要

如果您使用默认 zone 和 zone group 存储数据，则不要删除默认区域及其池。另外，删除默认 zone group 会删除系统用户。

要访问 **default** zone 和 zonegroup 中的旧数据，请在 **radosgw-admin** 命令中使用 **--rgw-zone default** 和 **--rgw-zonegroup default**。

示例

```
[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. 创建系统用户：

语法

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

示例

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

记录 **access_key** 和 **secret_key**。

f. 在主区中添加 access key 和 system key：

语法

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

示例

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

g. 提交更改：

语法

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

示例

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

h. 在 **cephadm** shell 外部，获取存储集群的 **FSID** 及进程：

示例

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

i. 启动 Ceph 对象网关守护进程：

语法

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

示例

```
[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. 在 Cephadm shell 中，配置 second zone。

a. 从主机拉取主要域配置：

语法

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

示例

```
[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. 从主机拉取主要 period 配置：

语法

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

示例

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

c. 配置 second zone:

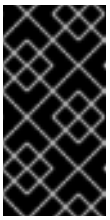
语法

```
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]
```

示例

```
[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. 可选：删除默认区。



重要

如果您使用默认 zone 和 zone group 存储数据，则不要删除默认区域及其池。

要访问 **default** zone 和 zonegroup 中的旧数据，请在 **radosgw-admin** 命令中使用 **--rgw-zone default** 和 **--rgw-zonegroup default**。

示例

```
[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. 更新 Ceph 配置数据库：

语法

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

示例

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

f. 提交更改：

语法

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

示例

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

g. 在 Cephadm shell 外，获取存储集群的 FSID 及进程：

示例

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

h. 启动 Ceph 对象网关守护进程：

语法

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

示例

```
[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. 可选：使用放置规格部署多站点 Ceph 对象网关：

语法

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

示例

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

验证

- 检查同步状态以验证部署：

  示例

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

3.5. 使用 CEPH 编排器移除 CEPH 对象网关

您可以使用 **ceph orch rm** 命令移除 Ceph 对象网关守护进程。

先决条件

- 一个正在运行的 Red Hat Ceph Storage 集群。
- 所有节点的根级别访问权限。
- 主机添加到集群中。
- 主机上至少部署了一个 Ceph 对象网关守护进程。

流程

1. 登录到 Cephadm shell：

   示例

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

2. 列出服务：

   示例

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

3. 删除服务：

   语法

   ```
   ceph orch rm SERVICE_NAME
   ```

   示例

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

验证

- 列出主机、守护进程和进程：

  语法

  ```
  ceph orch ps
  ```

  示例

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

其它资源

- 如需更多信息，请参阅 *Red Hat Ceph Storage Operations 指南中的使用命令行界面部分部署 Ceph 对象网关*。
- 如需更多信息，*请参阅 Red Hat Ceph Storage Operations 指南中的使用服务规格部署* Ceph 对象网关部分。

3.6. 使用 CEPH MANAGER RGW 模块

作为存储管理员，您可以使用 **rgw** 模块部署 Ceph 对象网关、单一站点和多站点。它有助于引导和配置 Ceph 对象域、zonegroup 和不同的相关实体。

您可以将可用令牌用于新创建的域或现有域。此令牌是一个 base64 字符串，封装 realm 信息及其 master zone 端点身份验证数据。

在多站点配置中，这些令牌可用于拉取域，以使用 **rgw zone create** 命令在主集群中与 master zone 同步的不同集群中创建 second zone。

3.6.1. 使用 rgw 模块部署 Ceph 对象网关

Bootstrap Ceph 对象网关域创建新的域实体、新 zonegroup 和新区域。**rgw** 模块指示编配器创建和部署对应的 Ceph 对象网关守护进程。

使用 **ceph mgr module enable rgw** 命令启用 **rgw** 模块。启用 **rgw** 模块后，可传递命令行中的参数，或者使用 **yaml** 规格文件来引导域。

先决条件

- 正在运行的 Red Hat Ceph Storage 集群部署至少一个 OSD。

流程

1. 登录到 Cephadm shell：

   示例

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

2. 启用 ` rgw` 模块：

   示例

   ■

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

3. 使用命令行或 yaml 规格文件引导 Ceph 对象网关域：

- 选项 1：使用命令行界面：

  语法

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

  示例

  ```
  [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.
  ```

- 选项 2：使用 yaml 规格文件：

  a. 作为 root 用户，创建 yaml 文件：

  语法

  ```
  rgw_realm: REALM_NAME
  rgw_zonegroup: ZONEGROUP_NAME
  rgw_zone: ZONE_NAME
  placement:
    hosts:
     - _HOSTNAME_1_
     - _HOSTNAME_2_
  ```

  示例

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

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

  b. 可选：您可以在 realm bootstrap 期间将 hostname 参数添加到 zonegroup 中：

  语法

  ```
  service_type: rgw
  ```

```
placement:
  hosts:
  - _host1_
  - _host2_
spec:
  rgw_realm: my_realm
  rgw_zonegroup: my_zonegroup
  rgw_zone: my_zone
  zonegroup_hostnames:
  - _hostname1_
  - _hostname2_
```

示例

```
service_type: rgw
placement:
  hosts:
  - _host1_
  - _host2_
spec:
  rgw_realm: my_realm
  rgw_zonegroup: my_zonegroup
  rgw_zone: my_zone
  zonegroup_hostnames:
  - foo
  - bar
```

c. **将 YAML 文件挂载到容器中的一个目录下：**

示例

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

d. **引导域：**

示例

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

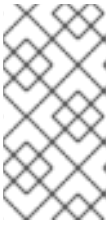
注意

rgw 模块使用的规格文件的格式与编配器使用的格式相同。因此，您可以提供任何编排支持的 Ceph 对象网关参数，包括 SSL 证书等高级配置功能。

4. **列出可用的令牌：**

示例

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

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



注意

如果在部署 Ceph 对象网关守护进程前运行上述命令，它会显示一个消息，因为还没有端点。

验证

- 验证对**象网关部署：**

示例

```
[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
```

- **通过 realm bootstrap 验证添加 的主机名 ：**

语法

```
radosgw-admin zonegroup get --rgw-zonegroup _zone_group_name_
```

示例

```
[ceph: root@host01 /]# radosgw-admin zonegroup get --rgw-zonegroup my_zonegroup

{
    "id": "02a175e2-7f23-4882-8651-6fbb15d25046",
    "name": "my_zonegroup_ck",
    "api_name": "my_zonegroup_ck",
    "is_master": true,
    "endpoints": [
        "http://vm-00:80"
    ],
    "hostnames": [
        "foo"
        "bar"
    ],
    "hostnames_s3website": [],
    "master_zone": "f42fea84-a89e-4995-996e-61b7223fb0b0",
```

```
    "zones": [
        {
            "id": "f42fea84-a89e-4995-996e-61b7223fb0b0",
            "name": "my_zone_ck",
            "endpoints": [
                "http://vm-00:80"
            ],
            "log_meta": false,
            "log_data": false,
            "bucket_index_max_shards": 11,
            "read_only": false,
            "tier_type": "",
            "sync_from_all": true,
            "sync_from": [],
            "redirect_zone": "",
            "supported_features": [
                "compress-encrypted",
                "resharding"
            ]
        }
    ],
    "placement_targets": [
        {
            "name": "default-placement",
            "tags": [],
            "storage_classes": [
                "STANDARD"
            ]
        }
    ],
    "default_placement": "default-placement",
    "realm_id": "439e9c37-4ddc-43a3-99e9-ea1f3825bb51",
    "sync_policy": {
        "groups": []
    },
    "enabled_features": [
        "resharding"
    ]
}
```


有关 Ceph 对象网关规格文件中的 zonegroup _ hostnames 中指定的主机名列表，请参阅 zonegroup 的 hostname 部分。

3.6.2. 使用 rgw 模块部署 Ceph 对象网关多站点

Bootstrap Ceph 对象网关域创建新的域实体、新 zonegroup 和新区域。它配置一个新的系统用户，可用于多站点同步操作。rgw 模块指示编配器创建和部署对应的 Ceph 对象网关守护进程。

使用 ceph mgr module enable rgw 命令启用 rgw 模块。启用 rgw 模块后，可传递命令行中的参数，或者使用 yaml 规格文件来引导域。

先决条件

- 正在运行的 Red Hat Ceph Storage 集群部署至少一个 OSD。

流程

1. 登录到 Cephadm shell：

   示例

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

2. 启用 ` rgw` 模块：

   示例

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

3. 使用命令行或 yaml 规格文件引导 Ceph 对象网关域：

   - 选项 1：使用命令行界面：

     语法

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

示例

```
[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.
```

- 选项 **2：使用 yaml 规格文件：**

a. **作为 root 用户，创建 yaml 文件：**

语法

```
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
```

示例

```
[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. **将 YAML 文件挂载到容器中的一个目录下：**

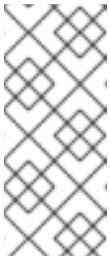
示例

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

c. **引导域：**

示例

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



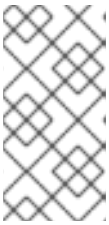
注意

rgw 模块使用的规格文件的格式与编配器使用的格式相同。因此，您可以提供任何编排支持的 Ceph 对象网关参数，包括 SSL 证书等高级配置功能。

4. **列出可用的令牌：**

示例

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

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



注意

如果在部署 Ceph 对象网关守护进程前运行上述命令，它会显示一个消息，因为还没有端点。

5. **使用以下令牌创建 second zone 并加入现有域：**

   a. **作为 root 用户，创建 yaml 文件：**

示例

```
[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. **将 zone-spec.yaml 文件挂载到容器中的一个目录中：**

示例

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

c. **在 second zone 中启用' rgw'module:**

示例

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

d. **创建 second zone：**

示例

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

验证

- 验证对**象网关多站点部署：**

示例

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

第 4 章 基本配置

作为存储管理员，了解配置 Ceph 对象网关的基础知识非常重要。您可以了解默认值和名为 Beast 的嵌入式 Web 服务器。若要对 Ceph 对象网关问题进行故障排除，您可以调整 Ceph 对象网关生成的日志记录和调试输出。另外，您可以使用 Ceph 对象网关为存储集群访问提供高可用性代理。

先决条件

- 一个运行良好、健康的 Red Hat Ceph Storage 集群。
- 安装 Ceph 对象网关软件包.

4.1. 为 DNS 添加通配符

您可以将通配符（如 hostname）添加到 DNS 服务器的 DNS 记录中。

前提条件

- 一个正在运行的 Red Hat Ceph Storage 集群。
- 安装了 Ceph 对象网关.
- 管理节点的根级别访问权限。

流程

1. 要将 Ceph 与 S3 样式子域搭配使用，请将通配符添加到 ceph-radosgw 守护进程用于解析域名的 DNS 服务器的 DNS 记录中：

   语法

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

对于 **dnsmasq，使用主机名前的句点(.)添加以下地址设置：**

语法

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

示例

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

对于 **bind，在 DNS 记录中添加通配符：**

示例

```
$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.

重启 DNS 服务器并使用子域对服务器发出 ping 命令，以确保 ceph-radosgw 守护进程可以处理子域请求：

语法

```
ping mybucket.HOSTNAME
```

示例

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

3. 如果 DNS 服务器位于本地机器上，则可能需要通过为本地机器添加 nameserver 条目来修改 /etc/resolv.conf。

4. 在 Ceph 对象网关 zone group 中添加主机名：

   a. 获取 zone group：

   语法

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

   示例

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

b. **备份 JSON 文件：**

示例

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

c. **查看 zonegroup.json 文件：**

示例

```
[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.

使用新的主机名更新 zonegroup.json 文件：

示例

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

e.

在 Ceph 对象网关中重新设置 zone group：

语法

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

示例

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

f. **更新周期：**

示例

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

g. **重新启动 Ceph 对象网关，以使 DNS 设置生效。**

其它资源

- **如需了解更多详细信息，请参阅 *Red Hat Ceph Storage 配置指南中的 Ceph 配置数据库部分。* *https://access.redhat.com/documentation/zh-cn/red_hat_ceph_storage/7/html-single/configuration_guide/#the-ceph-configuration-database_conf***

4.2. BEAST 前端 WEB 服务器

Ceph 对象网关提供 Beast，即 C/C 嵌入式前端 Web 服务器。Beast 使用 'Boost.Beast' C 库来解析 HTTP， Boost.Asio 用于异步网络 I/O。

其它资源

- ***Boost C++ Libraries***

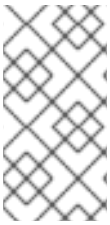
4.3. BEAST 配置选项

以下 Beast 配置选项可以传递到 RADOS 网关的 Ceph 配置文件中的嵌入式 Web 服务器。每个选项都有一个默认值。如果没有指定值，则默认值为空。

选项	描述	默认
endpoint 和 **ssl_endpoint**	以 **address[:port]** 格式设置监听地址，其中地址是以点十进制形式组成的 IPv4 地址字符串，或者用方括号括起的十六进制表示法中的 IPv6 地址。可选的端口默认为 **8080**（**端点**）和 **443**（**ssl_endpoint**）。它可以多次指定，如 **endpoint=[::1] endpoint=192.168.0.100:8000** 中。	EMPTY
ssl_certificate	用于启用 SSL 端点的 SSL 证书文件的路径。	EMPTY
ssl_private_key	用于启用 SSL 的端点的私钥文件的可选路径。如果未指定文件，则将使用 **ssl_certificate** 指定的文件作为私钥。	EMPTY
tcp_nodelay	在某些环境中进行性能优化。	EMPTY

使用 SSL 的带有 Beast 选项的 /etc/ceph/ceph.conf 文件示例：

```
...

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



注意

默认情况下，Beast 前端写入访问日志行，将服务器处理的所有请求记录到 RADOS 网关日志文件中。

其它资源

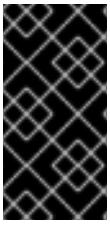
- ***如需更多信息，请参阅使用 Beast 前端。***

4.4. 为 BEAST 配置 SSL

您可以将 Beast 前端 web 服务器配置为使用 OpenSSL 库来提供传输层安全性(TLS)。若要将安全套接字层(SSL)与 Beast 搭配使用，您需要从与 Ceph 对象网关节点的主机名匹配的证书颁发机构(CA)获取证书。Beast 还需要 secret 密钥、服务器证书以及 single .pem 文件中的任何其他 CA。

重要

防止未经授权访问 .pem 文件，因为它包含 secret 密钥哈希。



重要

红帽建议从带有 Subject Alternative Name(SAN)字段的 CA 获取证书，以及用于 S3 样式子域的通配符。

重要

红帽建议仅将 SSL 与 Beast 前端 Web 服务器用于中小型测试环境。对于生产环境，您必须使用 HAProxy 和 keepalived 终止 HAProxy 中的 SSL 连接。

如果 Ceph 对象网关充当客户端，并且服务器上使用了自定义证书，您可以通过在节点上导入自定义 CA 来注入自定义 CA，然后使用 Ceph 对象网关规格文件中的 extra_container_args 参数将 etc/pki 目录映射到容器。

先决条件

- 一个运行良好、健康的 Red Hat Ceph Storage 集群。
- 安装 Ceph 对象网关软件包.
- 安装 OpenSSL 软件包.
- Ceph 对象网关节点的根级别访问权限.

流程

1. 在当前目录中创建一个名为 rgw.yml 的新文件：

示例

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

2. **打开 rgw.yml 文件进行编辑，并根据环境自定义该文件：**

语法

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

示例

```
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. ***使用服务规格文件部署 Ceph 对象网关：***

示例

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

4.5. D3N 数据缓存

datacenter-Data-Delivery Network (D3N)使用高速存储（如 NVMe ）在访问端缓存数据集。这种缓存允许大型数据作业使用边缘每个 Rados 网关节点上可用的计算和快速存储资源。Rados 网关充当后端对象存储(OSD)的缓存服务器，将数据存储在本地以供重复使用。

注意

每次重启 Rados 网关时，会清除缓存目录的内容。

4.5.1. 添加 D3N 缓存目录

要在 RGW 上启用 D3N 缓存，您需要在 podman unit.run 中包含 D3N 缓存目录。

先决条件

- ***一个正在运行的 Red Hat Ceph Storage 集群。***
- ***安装了 Ceph 对象网关.***

- *管理节点的根级别访问权限。*
- *每个 RGW 节点中有一个快速 NVMe 驱动器，用作本地缓存存储。*

流程

1. *为 NVMe 驱动器创建挂载点。*

   语法

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

   示例

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

2. *创建缓存目录路径。*

   语法

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

   示例

   -

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

3. **向 nvme-mount-path 和 rgw_d3n_l1_datacache_persistent_path 提供 +rwx 权限。**

语法

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

示例

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

4. **使用 extra_container_args 创建/修改 RGW 规范文件，将 rgw_d3n_l1_datacache_persistent_path 添加到 podman unit.run 中。**

语法

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

示例

```
[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/"
```

注意

如果单个主机上有多个 RGW 实例，则必须为每个实例创建一个单独的 rgw_d3n_l1_datacache_persistent_path，并在 extra_container_args 中添加每个路径。

示例 ：

对于每个主机中的两个 RGW 实例，在 rgw_d3n_l1_datacache_persistent_path 下创建两个单独的 cache-directory ：/mnt/nvme0n1/rgw_datacache/rgw1 和 /mnt/nvme0n1/rgw_datacache/rgw2

rgw 规范文件中的 "extra_container_args" 示例 ：

```
"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 示例 ：

```
[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. 重新部署 RGW 服务：

示例

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

4.5.2. 在 rados 网关上配置 D3N

您可以在现有 RGW 上配置 D3N 数据缓存，以提高 Red Hat Ceph Storage 集群中运行的 big-data 作业性能。

先决条件

- 一个正在运行的 Red Hat Ceph Storage 集群。
- 安装了 Ceph 对象网关.
- 管理节点的根级别访问权限。
- 一个快速的 NVMe 作为缓存存储。

添加所需的 D3N 相关配置

要在现有 RGW 上启用 D3N，需要为每个 Rados 网关客户端设置以下配置：

语法

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

- ***rgw_d3n_l1_local_datacache_enabled=true***
- ***rgw_d3n_l1_datacache_persistent_path= 路径到缓存目录***

示例

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

- ***rgw_d3n_l1_datacache_size=max_size_of_cache_in_bytes***

示例

```
rgw_d3n_l1_datacache_size=10737418240
```

示例流程

1. *创建 **test** 对象：*

注意

*测试对象需要大于 **4 MB** 才能缓存。*

示例

```
[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. 执行对象的 GET ：

示例

```
[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. 验证缓存创建。将创建缓存，其名称由配置的 rgw_d3n_l1_datacache_persistent_path 中的对象 key-name 组成。

示例

```
[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. 为对象创建缓存后，该对象的下一个 GET 操作将从缓存访问，从而加快访问速度。

示例

```
[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
```

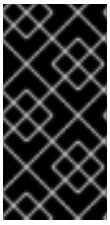
在上例中，为了演示缓存加速，我们正写入 RAM 驱动器(/dev/shm)。

其它资源

- 有关使用高可用性的更多详细信息，请参阅 Red Hat Ceph Storage 故障排除指南中的 Ceph 子系统默认日志记录级别值 部分。
- 有关使用高可用性的更多详细信息，请参阅 Red Hat Ceph Storage 故障排除指南中的 了解 Ceph 日志 部分。

4.6. 调整日志记录和调试输出

完成设置过程后，检查日志输出以确保它满足您的需要。默认情况下，Ceph 守护进程日志到 journald，您可以使用 journalctl 命令查看日志。或者，您也可以将 Ceph 守护进程日志指向文件，这些文件位于 /var/log/ceph/CEPH_CLUSTER_ID/ 目录下。



重要

详细日志记录每小时可生成超过 1 GB 数据。这种类型的日志记录可能会填满操作系统的磁盘，从而导致操作系统停止正常运行。

先决条件

- 一个正在运行的 Red Hat Ceph Storage 集群。
- 安装 Ceph 对象网关软件.

流程

1. 设置以下参数以增加 Ceph 对象网关日志输出：

语法

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

示例

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

a. **您还可以在运行时修改这些设置：**

语法

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

示例

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

2. **（可选）您可以将 Ceph 守护进程配置为将其输出记录到文件中。将 log_to_file 和 mon_cluster_log_to_file 选项设置为 true ：**

示例

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

其它资源

- 如需了解更多详细信息，请参阅 Red Hat Ceph Storage 配置指南中的 Ceph 调试和日志记录配置部分。 https://access.redhat.com/documentation/zh-cn/red_hat_ceph_storage/7/html-single/configuration_guide/#ceph-debugging-and-logging-configuration

4.7. 静态 WEB 托管

作为存储管理员，您可以将 Ceph 对象网关配置为托管 S3 存储桶中的静态网站。传统网站托管涉及为各个网站配置 Web 服务器，当内容未动态更改时，此服务器会以低效的方式使用资源。例如，站点不使用 PHP、servlets、database、nodejs 等服务器端服务。这种方法比设置具有每个站点 Web 服务器的虚拟机更经济。

先决条件

- 一个正常运行的 Red Hat Ceph Storage 集群。

4.7.1. 静态 Web 托管假设

静态 Web 托管至少需要一个运行 Red Hat Ceph Storage 集群，以及至少两个用于静态网站的 Ceph 对象网关实例。红帽假定每个区域都将具有多个使用负载平衡器的网关实例，如高可用性(HA)代理和 keepalived。

重要

红帽 不支持 使用 Ceph 对象网关实例来同时部署标准 S3/Swift API 和静态 Web 主机。

其它资源

- 有关使用高可用性的更多详细信息，请参阅 Red Hat Ceph Storage 对象网关指南中的 高可用性服务部分。 https://access.redhat.com/documentation/zh-cn/red_hat_ceph_storage/7/html-single/object_gateway_guide/#high-availability-service_rgw

4.7.2. 静态 Web 托管要求

静态 Web 托管功能使用自己的 API，因此将网关配置为在 S3 存储桶中使用静态网站需要以下内容：

1. S3 静态 Web 托管使用 Ceph 对象网关实例，这些实例与用于标准 S3/Swift API 用例的实例不同。

2. 托管 S3 静态网站的网关实例应具有独立于标准 S3/Swift API 网关实例的单独、非覆盖域名。

3. 托管 S3 静态网站的网关实例应使用与标准 S3/Swift API 网关实例独立的面向公共的 IP 地址。

4. 托管 S3 静态 web 站点负载均衡的网关实例，如有必要，使用 HAProxy/keepalived 终止 SSL。

4.7.3. 静态 Web 托管网关设置

要为静态 Web 托管启用 Ceph 对象网关，请设置以下选项：

语法

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

示例

```
[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 设置需要为 true。rgw_enable_apis 设置需要启用 s3website API。rgw_dns_name 和 rgw_dns_s3website_name 设置必须提供其完全限定的域。如果站点使用规范名称扩展，则将 rgw_resolve_cname 选项设置为 true。

重要

rgw_dns_name 和 rgw_dns_s3website_name 的 FQDN 不能 重叠。

4.7.4. 静态 Web 托管 DNS 配置

以下是假定 DNS 设置的示例，其中前两行使用标准 S3 接口指定网关实例的域，并指向 IPv4 和 IPv6 地址。第三行使用规范名称扩展为 S3 存储桶提供通配符 CNAME 设置。第四行和第五个行使用 S3 网站接口指定网关实例的域，并指向其 IPv4 和 IPv6 地址。

```
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
```

注意

前两行中的 IP 地址与第四和第五个行中的 IP 地址有所不同。

如果在多站点配置中使用 Ceph 对象网关，请考虑使用路由解决方案将流量路由到最接近客户端的网关。

Amazon Web Service(AWS)需要静态 Web 主机存储桶才能与主机名匹配。Ceph 提供了几种不同的配置 DNS 的方式，如果代理具有匹配的证书，HTTPS 将正常工作。

Subdomain 中 Bucket 的主机名

要使用 AWS 风格的 S3 子域，请在 DNS 条目中使用通配符，该条目可将请求重定向到任何存储桶。DNS 条目可能类似如下：

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

使用以下方法访问存储桶名称为 bucket1 的存储桶名称：

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

主机名到非匹配问题

Ceph 支持将域名映射到 bucket，而不在请求中包含 bucket 名称，这对 Ceph 对象网关而言是唯一的。要使用域名访问 bucket，请将域名映射到 bucket 名称。DNS 条目可能类似如下：

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

存储桶名称为 bucket2。

使用以下方法访问存储桶：

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

使用 CNAME 到 Long Bucket 的主机名

AWS 通常需要存储桶名称来匹配域名。要使用 CNAME 为静态 Web 托管配置 DNS，DNS 条目可能类似如下：

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

使用以下方法访问存储桶：

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

没有 CNAME 的 Long Bucket 的主机名

如果 DNS 名称包含其他非CNAME 记录，如 SOA、NSX 或 TXT，DNS 记录必须将域名直接映射到 IP 地址。例如：

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

使用以下方法访问存储桶：

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

4.7.5. 创建静态 Web 托管站点

要创建静态网站，请执行以下步骤：

1. 创建 S3 存储桶。bucket 名称 MIGHT 与网站的域名相同。例如，mysite.com 可能具有 bucket 名称 mysite.com。AWS 需要此功能，但 Ceph 不需要它。

   - 详情请参阅 Red Hat Ceph Storage 对象网关指南中的 静态 Web 托管 DNS 配置部分。

2. 将静态网站内容上传到 bucket。内容可能包括 HTML、CSS、客户端 JavaScript、图像、音频/视频内容和其他可下载的文件。网站 MUST 有一个 index.html 文件，可能有一个 error.html 文件。

3. 验证网站内容。此时，只有存储桶的创建者有权访问其内容。

4. 设置文件的权限，以便可以公开读取。

4.8. CEPH 对象网关的高可用性

作为存储管理员，您可以将多个 Ceph 对象网关实例分配到一个区域。这可让您随着负载增加（即同一 zone group 和 zone）进行横向扩展，但您不需要联合架构来使用高可用性代理。由于每个 Ceph 对象网关守护进程都有自己的 IP 地址，因此您可以使用 ingress 服务在许多 Ceph 对象网关守护进程或节点之间平衡负载。ingress 服务管理 Ceph 对象网关环境的 HAProxy 和 keepalived 守护进程。您还可以在 HAProxy 服务器上终止 HTTPS 流量，并为 Ceph 对象网关使用 HAProxy 服务器和 Beast 前端 web 服务器实例之间的 HTTP。

先决条件

- 在不同主机上运行至少两个 Ceph 对象网关守护进程。
- 具有不同主机上运行的 入口 服务的两个实例的容量。

4.8.1. 高可用性服务

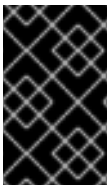
ingress 服务为 Ceph 对象网关提供高可用性端点。可以根据需要将入口 服务部署到任意数量的主机上。红帽建议至少有两个支持的 Red Hat Enterprise Linux 服务器，每个服务器都配置了 ingress 服务。您可以使用最小配置选项运行高可用性(HA)服务。Ceph 编配器部署 入口 服务，该服务通过提供浮

动虚拟 IP 地址的负载平衡来管理 haproxy 和 keepalived 守护进程。活跃的 haproxy 将所有 Ceph 对象网关请求分发到所有可用的 Ceph 对象网关守护进程。

一次在其中一个 入口 主机上自动配置虚拟 IP 地址，称为主主机。Ceph 编配器基于配置为同一子网一部分的现有 IP 地址选择第一个网络接口。如果虚拟 IP 地址不属于同一子网，您可以为 Ceph 编配器定义子网列表以匹配现有 IP 地址。如果 keepalived 守护进程和活跃 haproxy 在主主机上没有响应，则虚拟 IP 地址会移到备份主机上。此备份主机成为新的主主机。

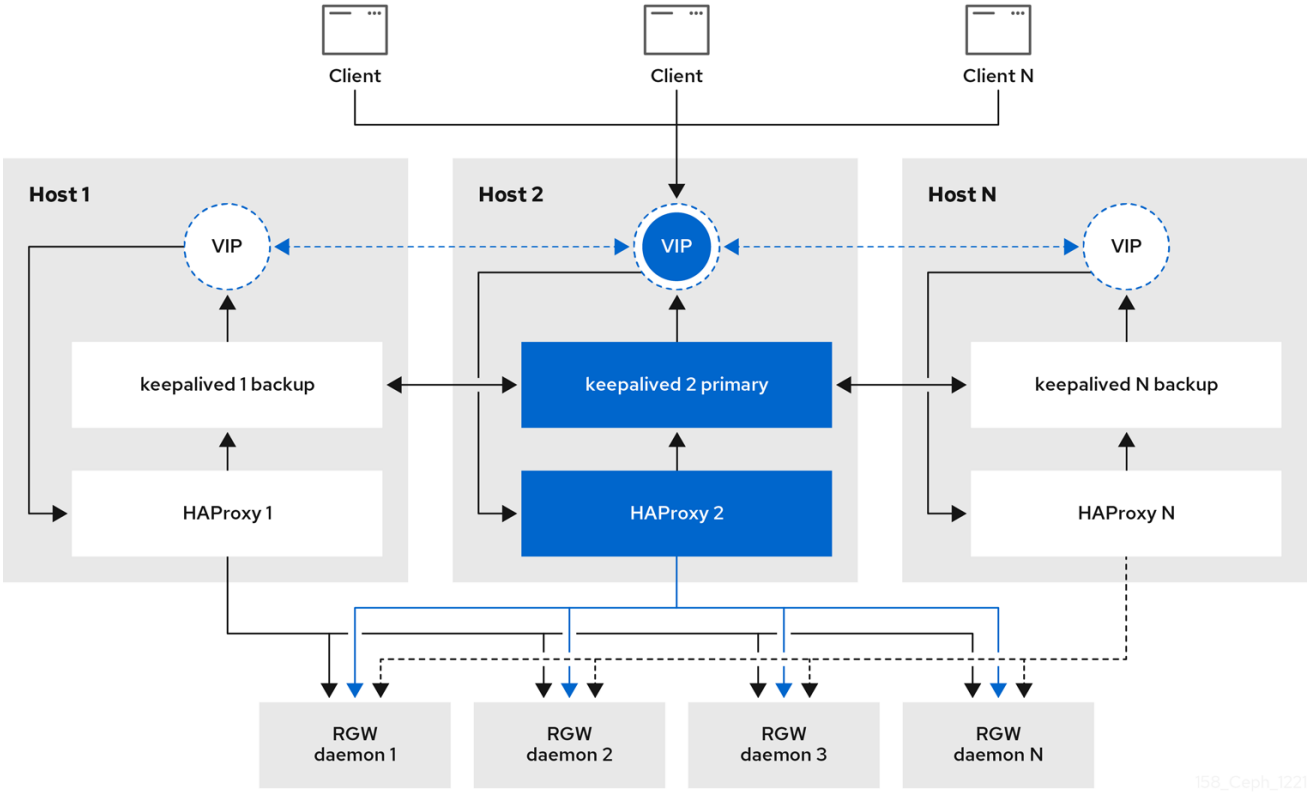
警告

目前，您无法在没有配置 IP 地址的网络接口中配置虚拟 IP 地址。



重要

要使用安全套接字层(SSL)，SSL 必须通过 ingress 服务而不是 Ceph 对象网关终止。



4.8.2. 为 Ceph 对象网关配置高可用性

要为 Ceph 对象网关配置高可用性(HA)，您编写 YAML 配置文件，Ceph 编配器则执行 入口 服务的安装、配置和管理。ingress 服务使用 haproxy 和 keepalived 守护进程为 Ceph 对象网关提供高可用性。

先决条件

- 至少两个运行 Red Hat Enterprise Linux 9 或更高版本的主机，用于安装 ingress 服务。
- 一个正常运行的 Red Hat Ceph Storage 集群。
- 最少两个 Ceph 对象网关守护进程在不同主机上运行。
- 对运行 入口 服务的主机的根级别访问权限。
- 如果使用防火墙，则打开 HTTP 的端口 80，为 HTTPS 流量打开端口 443。

流程

1. 创建新的 ingress.yaml 文件：

示例

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

2. 打开 ingress.yaml 文件进行编辑。添加了以下选项，并添加适用于环境的值：

语法

```
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) 必须设置为 **ingress**。

(2) 必须与现有的 **Ceph** 对象网关服务名称匹配。

(3) 部署 **haproxy** 和 **keepalived** 容器的位置。

(4) **ingress** 服务可用的虚拟 **IP** 地址。

(5) 访问入口服务的端口。

(6) 访问 **haproxy** 负载均衡器状态的端口。

(7) 可用子网的可选列表。

(8) 可选 **SSL** 证书和私钥。

示例

```
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. 启动 **Cephadm shell：**

示例

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

4.

配置最新的 haproxy 和 keepalived 镜像：

语法

```
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.

使用 Ceph 编配器安装和配置新的 ingress 服务：

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

6.

Ceph 编配器完成后，验证 HA 配置。

a.

在运行 ingress 服务的主机上，检查是否显示虚拟 IP 地址：

示例

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

'

b. 尝试从 Ceph 客户端访问 Ceph 对象网关：

语法

```
wget HOST_NAME
```

示例

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

如果返回带有类似以下示例内容的 index.html，则 Ceph 对象网关的 HA 配置可以正常工作。

示例

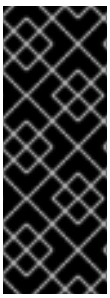
```
<?xml version="1.0" encoding="UTF-8"?>
 <ListAllMyBucketsResult xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
  <Owner>
   <ID>anonymous</ID>
   <DisplayName></DisplayName>
  </Owner>
  <Buckets>
  </Buckets>
 </ListAllMyBucketsResult>
```

其他资源

- 如需了解更多详细信息，请参阅 执行标准 RHEL 安装指南。

- 如需了解更多详细信息，请参阅 Red Hat Ceph Storage 对象网关指南中的 高可用性服务 部分。

4.9. 使用 CEPH 对象网关配置 NFS

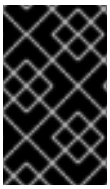


重要

使用 Ceph 对象存储后端的 NFS 不是全面的 NFS 服务。它的主要目的是，通过 NFS 文件系统优化数据，对传统应用进行无缝迁移，以使用文件与 Ceph 对象存储进行对象存储。然后，通过 S3 端点作为 S3 存储桶访问数据。对于具有高可用性和透明故障转移等功能的完整 NFS 解决方案，您应该将 NFS 与 CephFS 后端搭配使用。

NFS 服务使用 Cephadm 通过 Ceph 对象存储后端部署。NFS 的配置存储在 nfs-ganesha 池中，导出通过命令行界面(CLI)命令和 Ceph 控制面板进行管理。如需更多信息，请参阅使用 Ceph 对象存储后端部署 NFS 服务，将命名空间导出到 NFS-Ganesha，以及管理 NFS Ganesha 导出。

Ceph 对象网关命名空间可以通过基于文件的 NFSv4 协议导出，以及传统的 HTTP 访问协议(S3 和 Swift)。特别是，现在可将 Ceph 对象网关配置为在 NFS-Ganesha NFS 服务器中嵌入时提供基于文件的访问。



重要

在使用基于 Cephadm 或 Rook 的部署时，只支持 NFSv4 协议。

命名空间约定

NFS 符合 Amazon Web Services (AWS)层次结构命名空间惯例，将 UNIX 样式的路径名称映射到 S3 存储桶和对象。

附加命名空间的顶级由 S3 存储桶组成，以 NFS 目录表示。文件和目录（对应于 bucket）各自代表为对象、遵循 S3 前缀和分隔符惯例。/ 是唯一受支持的路径分隔符。

例如，如果 NFS 客户端已将 RGW 命名空间挂载到 /nfs，那么 NFS 命名空间中的文件 /nfs/mybucket/www/index.html 对应于 bucket/container mybucket 中的 RGW 对象 www/index.html。

支持的操作的限制

Ceph Object Storage NFS 接口支持对文件和目录的大部分操作，其限制如下：

- 不支持链接，包括符号链接。
- 不支持 NFS ACL。
  - 支持 UNIX 用户和组所有权和权限。
- 目录可能无法移动/重新命名。
  - 文件可以在目录之间移动。
- 仅支持 full, sequential write I/O
  - 写入操作受限于 上传。
  - 许多典型的 I/O 操作（如编辑文件）将失败，因为它们执行非排序存储。
  - 有些文件实用程序按顺序编写（例如，一些 GNU tar 版本）可能会因为不常的非排序存储而失败。
  - 当通过 NFS 挂载时，顺序应用程序 I/O 通常可以通过同步挂载选项按顺序写入 NFS 服务器。例如： Linux 中的 -osync。
  - 无法同步挂载的 NFS 客户端（例如 MS Windows）将无法上传文件。

4.9.1. 将命名空间导出到 NFS-Ganesha

若要配置新的 NFS Ganesha 导出以用于 Ceph 对象网关，您必须使用 Red Hat Ceph Dashboard。如需了解更多详细信息，请参阅 Red Hat Ceph Storage 仪表板指南中的在 Ceph 仪表板中管理 NFS Ganesha 导出 部分。

重要

对于使用 Ceph 对象网关的现有 NFS 环境，目前不支持从 Red Hat Ceph Storage 4 升级到 Red Hat Ceph Storage 5。

重要

红帽仅支持使用 Ceph 对象网关导出 NFS 版本 4。

您只能使用命令行界面(CLI)创建用户级 NFS Ganesha 导出。

先决条件

- 正在运行的 Red Hat Ceph Storage
- 已创建的用户。如需更多信息，请参阅 创建用户。

流程

1. 登录 Cephadm shell。

语法

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

2. 在根目录中创建用户级导出。

语法

```
ceph nfs export create rgw --cluster-id NFS_CLUSTER_NAME --pseudo-path
PATH_FROM_ROOT --user-id USER_ID
```

示例

```
[ceph:root@host01 /]# ceph nfs export create rgw --cluster-id cluster1 --pseudo-path
root/testnfs1/ --user-id nfsuser
```

3. **挂载 NFS。**

语法

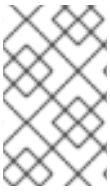
```
mount -t nfs IP_ADDRESS:PATH_FROM_ROOT -osync MOUNT_POINT
```

示例

```
[ceph:root@host01 /]# mount -t nfs 10.0.209.0:/root/testnfs1 -osync /mnt/mount1
```

重要

对于大型上传 >200 GB，使用 -osync 挂载可能会影响输入/输出操作。将 S3 与多部分搭配使用，以上传此类对象。



注意

如果您在存储桶上运行 setattr，它会静默阻止在代表存储桶的路径上设置属性。

第 5 章 多站点配置和管理

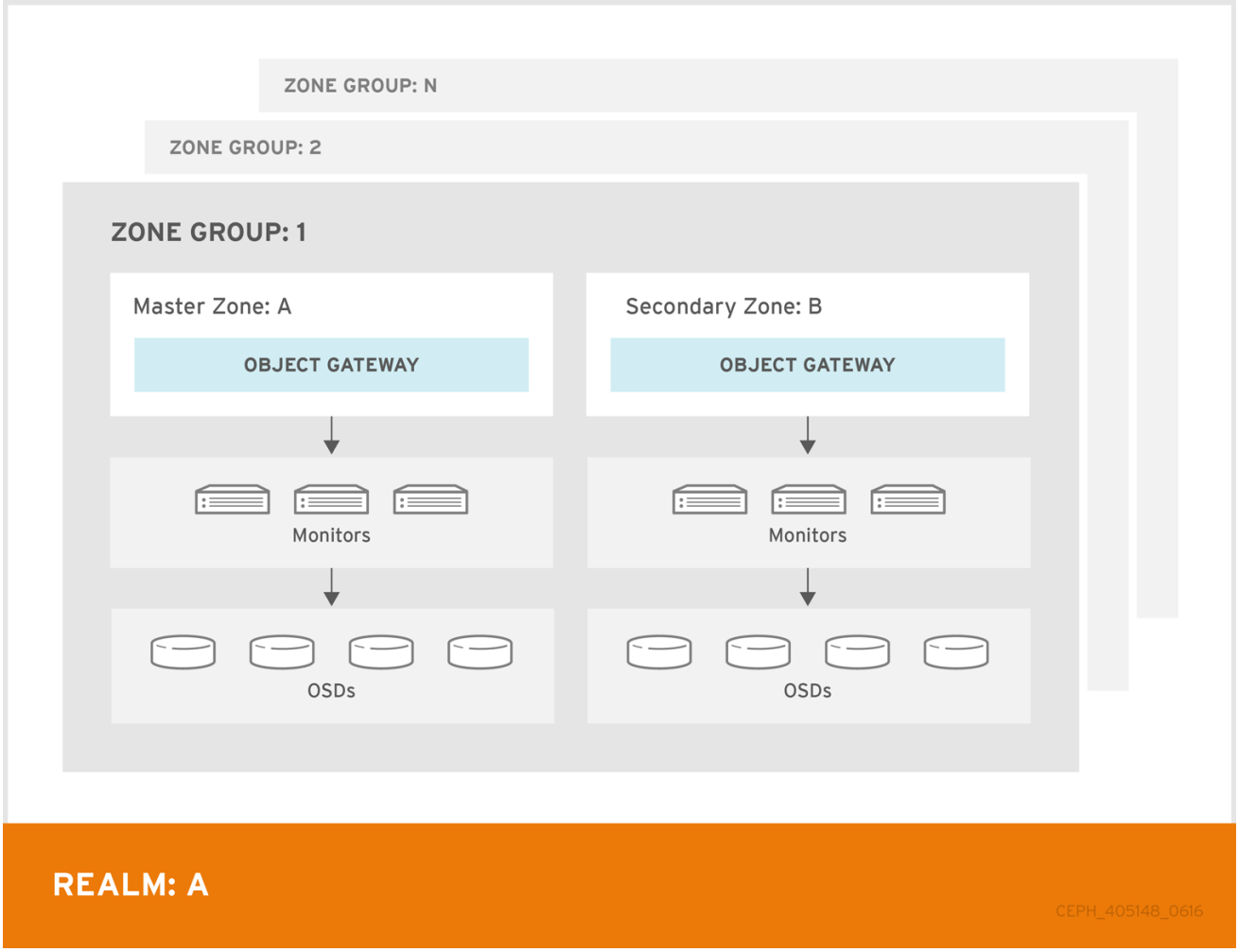
作为存储管理员，您可以为各种用例配置和管理多个 Ceph 对象网关。您可以了解在灾难恢复和故障转移事件期间要做什么。另外，您可以在多站点 Ceph 对象网关环境中了解更多有关 realms、zone 和 syncing 策略的信息。

单一区配置通常由一个 zone group 包含一个 zone，以及一个或多个 ceph-radosgw 实例，您可以在实例之间平衡网关客户端请求。在单一区域配置中，通常多个网关实例指向单个 Ceph 存储集群。但是，红帽支持 Ceph 对象网关的几个多站点配置选项：

- 多区：更高级的配置由一个 zone group 和多个 zone 组成，每个 zone 包含一个或多个 ceph-radosgw 实例。每个区域都由自己的 Ceph 存储集群支持。zone group 中的多个区域为 zone group 提供灾难恢复，当其中一个区遇到重大故障时。每个区域都处于活跃状态，并可能会接收写入操作。除了灾难恢复外，多个活动区域也可能充当内容交付网络的基础。要在没有复制的情况下配置多个区域，请参阅 配置多个区域而无需复制。

- multi-zone-group: Formerly called 'regions'，Ceph 对象网关也可以支持多个 zone group，每个 zone group 具有多个 zone。存储在同一域中的 zone group 的对象共享全局命名空间，确保 zone group 和 zone 之间的唯一对象 ID。

- 多个 Realms：Ceph 对象网关支持域的概念，可以是单个 zone group 或多个 zone group，以及域的全局唯一命名空间。多个域提供了支持大量配置和命名空间的功能。



先决条件

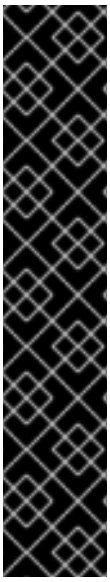
- *一个正常运行的 Red Hat Ceph Storage 集群。*
- *部署 Ceph 对象网关软件。*

5.1. 要求和假设

多站点配置至少需要两个 Ceph 存储集群，以及至少两个 Ceph 对象网关实例，每个 Ceph 存储集群一个。

本指南假定在地理上独立位置至少有两个 Ceph 存储集群，但配置可以在同一物理站点上工作。本指南还假设四个 Ceph 对象网关服务器分别名为 rgw1、rgw 2、rgw3 和 rgw4。

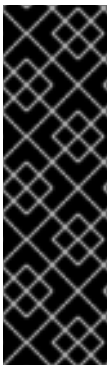
多站点配置需要一个 master zone group 和 master zone。此外，每个 zone group 需要 master zone。zone group 可能具有一个或多个次要或非 master 区域。



重要

在为多站点规划网络注意事项时，务必要了解多站点同步网络上观察的关系带宽和延迟，以及客户端与与次要站点同步对象的当前同步状态直接关联率。Red Hat Ceph Storage 多站点集群之间的网络链接必须能够处理到主集群，以便在次要站点上保持有效的恢复时间。多站点同步是异步的，其中一个限制是同步网关可在链接间处理数据的速率。对于每个客户端网关，每个 8 TB 或累积接收数据，例如，要查看网络互连速度的示例可以是 1GbE 或数据中心间连接。因此，如果您复制到两个其他站点，每天 16 TB，则需要 6 GbE 专用带宽用于多站点复制。

红帽还推荐使用私有以太网或 Dense wavelength-division 多路复位(DWDM)作为互联网的 VPN 并不理想，因为产生的额外开销并不是一个理想的选择。



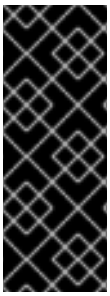
重要

realm 的 master zone group 中的 master zone 负责存储 realm 元数据的主副本，包括用户、配额和 bucket （由 radosgw-admin CLI 创建）。此元数据会自动同步到 second zone 和 second zone group。通过 radosgw-admin CLI 执行的元数据操作必须在 master zone group 的 master zone 中的主机上执行，以确保它们与 second zone group 和 zone 同步。目前，可以对二级域和域组执行元数据操作，但不建议这么做，因为它们不会被同步，从而导致元数据碎片。

注意

对于多站点中的新 Ceph 对象网关部署，需要大约 20 分钟时间才能将元数据操作同步到次要站点。

在以下示例中，rgw1 主机将充当 master zone group 的 master zone； rgw2 主机将充当 master zone group 的 second zone； rgw3 主机将充当 second zone group 的 master zone； rgw4 主机将充当 second zone group 的 second zone。



重要

红帽建议使用负载均衡器和三个 Ceph 对象网关守护进程与多站点同步端点。对于多站点配置中的非同步 Ceph 对象网关节点，这些节点专用于通过负载均衡器的客户端 I/O 操作，请运行 ceph config set client.rgw.CLIENT_NODE rgw_run_sync_thread false 命令，以防止它们执行同步操作，然后重启 Ceph 对象网关。

以下是 HAProxy 用于同步网关的典型配置文件：

示例

```
[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
```

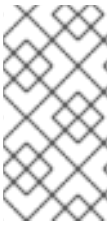
```
    stats enable
    stats hide-version
    stats refresh 30s
    stats show-node
##     stats auth admin:password
    stats uri  /haproxy?stats
     stats admin if TRUE


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. 池

红帽建议使用 Ceph Placement Group's per Pool Calculator 来计算要创建 radosgw 守护进程的池的放置组的适当数量。在 Ceph 配置数据库中将计算的值设置为默认值。

示例

```
[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
```



注意

在 Ceph 对象网关实例创建池时，对 Ceph 配置进行此更改将使用这些默认值。或者，您也可以手动创建池。

特定于区域的池名称遵循命名规则 ZONE_NAME.POOL_NAME。例如，名为 us-east 的区域将具有以下池：

- .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

其它资源

其他资源

- 有关创建 池的详细信息，请参阅 Red Hat Ceph Storage 策略指南中的 池章节。

5.3. 将单个站点系统迁移到多站点

要从带有 default zone 组和 zone 的单个站点系统迁移到多站点系统，请使用以下步骤：

1. 创建 realm。将 REALM_NAME 替换为域名称。

   语法

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

2. 重命名默认区域和 zonegroup。将 NEW_ZONE_GROUP_NAME 和 NEW_ZONE_NAME 替换为 zonegroup 和 zone 名称。

   语法

   ```
   radosgw-admin zonegroup rename --rgw-zonegroup default --zonegroup-new-name NEW_ZONE_GROUP_NAME
   radosgw-admin zone rename --rgw-zone default --zone-new-name NEW_ZONE_NAME --rgw-zonegroup NEW_ZONE_GROUP_NAME
   ```

3. 重命名默认 zonegroup 的 api_name。将 NEW_ZONE_GROUP_NAME 替换为 zonegroup 名称。

   语法

```
radosgw-admin zonegroup modify --api-name NEW_ZONE_GROUP_NAME --rgw-
zonegroup NEW_ZONE_GROUP_NAME
```

4. ***配置主要 zonegroup。将 NEW_ZONE_GROUP_NAME 替换为 zonegroup 名称，将 REALM_NAME 替换为域名。将 ENDPOINT 替换为 zonegroup 中的完全限定域名。***

   语法

```
radosgw-admin zonegroup modify --rgw-realm REALM_NAME --rgw-zonegroup
NEW_ZONE_GROUP_NAME --endpoints http://ENDPOINT --master --default
```

5. ***配置主区域。将 REALM_NAME 替换为域名，NEW_ZONE_GROUP_NAME 替换为 zonegroup 名称，NEW_ZONE_NAME 替换为区域名称，将 ENDPOINT 替换为 zonegroup 中的完全限定域名。***

   语法

```
radosgw-admin zone modify --rgw-realm REALM_NAME --rgw-zonegroup
NEW_ZONE_GROUP_NAME --rgw-zone NEW_ZONE_NAME --endpoints http://ENDPOINT
--master --default
```

6. ***创建系统用户。使用用户名替换 USER_ID。将 DISPLAY_NAME 替换为显示名称。它可以包含空格。***

   语法

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

7. ***提交更新的配置：***

示例

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

8. ***grep 用于 rgw 服务名称***

语法

```
ceph orch ls | grep rgw
```

9. ***设置 realm、zonegroup 和 primary zone 的配置。***

语法

```
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
```

示例

```
[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
```

10. ***重启 Ceph 对象网关：***

示例

```
[ceph: root@host01 /]# systemctl restart ceph-radosgw@rgw.`hostname -s`
```

语法

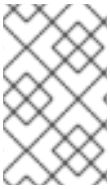
```
[ceph: root@host01 /]# ceph orch restart _RGW_SERVICE_NAME_
```

示例

```
[ceph: root@host01 /]# ceph orch restart rgw.rgwsvcid.mons-1.jwgwwp
```

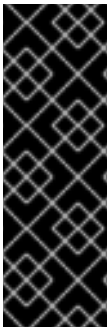
5.4. 建立二级 ZONE

zone group 中的 zone 复制所有数据，以确保每个区具有相同的数据。在创建二级 zone 时，在标识为服务二级 zone 的主机上执行所有 radosgw-admin zone 操作。



注意

要添加其他区域，其步骤与添加二级 zone 相同。使用不同的区名称。



重要

您必须在 master zonegroup 的 master zone 内运行元数据操作，如用户创建和配额。master zone 和 second zone 可以从 RESTful API 接收 bucket 操作，但 second zone 会将 bucket 操作重定向到 master zone。如果 master zone 为 down，则存储桶操作将失败。如果使用 radosgw-admin CLI 创建存储桶，您必须在 master zone group 的 master zone 中的主机上运行它，以便存储桶将与其他 zone group 和 zone 同步。

先决条件

- *至少两个正在运行的 Red Hat Ceph Storage 集群。*
- *至少两个 Ceph 对象网关实例，每个实例对应一个 Red Hat Ceph Storage 集群。*
- *所有节点的根级别访问权限。*
- *节点或容器添加到存储集群中。*
- *部署所有 Ceph 管理器、监控器和 OSD 守护进程。*

流程

1. *登录 cephadm shell：*

示例

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

2. ***从主机拉取主要域配置：***

语法

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

示例

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

3. ***从主机拉取主要 period 配置：***

语法

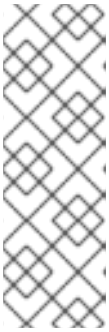
```
radosgw-admin period pull --url=URL_TO_PRIMARY_ZONE_GATEWAY --access-key=ACCESS_KEY --secret-key=SECRET_KEY
```

示例

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

4.

配置 second zone:



注意

所有区域默认在主动配置中运行；即，网关客户端可能会将数据写入任何区域，并且区域会将数据复制到 zone group 中所有其他 zone。如果 second zone 不应该接受写操作，请指定 '--read-only 标志，以在 master 区域和 second 区域之间创建主动 - 被动配置。另外，提供生成的系统用户的 access_key 和 secret_key 存储在 master zone group 的 master zone 中。

语法

```
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]
```

示例

```
[ceph: root@host04 /]# radosgw-admin zone create --rgw-zonegroup=us --rgw-zone=us-east-
2 --endpoints=http://rgw2:80 --access-key=LIPEYZJLTWXRKXS9LPJC --secret-
key=lsAje0AVDNXNw48LjMAimpCpI7VaxJYSnfD0FFKQ
```

5.

可选：删除默认区：

重要

如果您使用默认 zone 和 zone group 存储数据，则不要删除默认区域及其池。

示例

```
[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. **更新 Ceph 配置数据库：**

语法

```
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
```

示例

```
[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. ***提交更改：***

语法

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

示例

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

8. ***在 cephadm shell 外部，获取存储集群的 FSID 及进程：***

示例

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

9. *启动* ***Ceph*** *对象网关守护进程：*

语法

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

示例

```
[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. 配置归档区

使用 Object Storage Archive Zone 功能归档位于 Red Hat Ceph Storage 上的对象数据。

存档区域在 Ceph 对象网关中使用多站点复制和 S3 对象版本控制功能。archive 区域保留所有可用对象的所有版本，即使 production 文件中被删除也是如此。

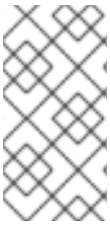
archive 区域具有 S3 对象的版本历史记录，只能通过与存档区域关联的网关删除。它捕获所有数据更新和元数据，将它们整合为 S3 对象的版本。

在创建存档区域后，可以使用 bucket 粒度复制到存档区域。

您可以通过存储桶生命周期策略控制存档区的存储空间使用，您可以在其中为对象定义您要保留的版本数量。

归档区域有助于保护您的数据不受逻辑或物理错误的影响。它可以从逻辑故障保存用户，如意外删除 production 区域中的存储桶。它还可从大量硬件故障中保存您的数据，如完整的生产站点故障。另外，它还提供了一个不可变的副本，可帮助构建可运行的应用程序保护策略。

要实现存储桶粒度复制，请使用 sync policy 命令启用和禁用策略。如需更多信息，请参阅创建同步策略组 和修改同步策略组。



注意

使用同步策略组流程是可选的，只需要在存储桶粒度复制中使用和禁用。要在没有存储桶粒度复制的情况下使用归档区域，不需要使用同步策略过程。

如果要从单一站点迁移存储集群，请参阅将单个站点系统迁移到多站点。

先决条件

- 一个正在运行的 Red Hat Ceph Storage 集群。
- Ceph 监控节点的根级别访问权限.
- 安装 Ceph 对象网关软件.

流程

- 在新区域创建过程中，使用 存档 层来配置存档区域。

语法

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

示例

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

- 从归档区修改存档区域，以仅从主区域同步并执行 period 更新提交。

语法

```
$ radosgw-admin zone modify --rgw-zone archive --sync_from primary --sync_from_all false
--sync-from-rm secondary

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

注意

建议是将 max_objs_per_shard 的 max_objs_per_shard 减小到 50K，以考虑归档区中的 omap olh 条目。这有助于为每个存储桶索引分片对象保留 omap 条目数量，以防止大型 omap 警告。

例如，

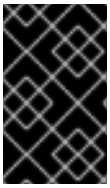
```
$ ceph config set client.rgw rgw_max_objs_per_shard 50000
```

其他资源

- 如需了解更多详细信息，请参阅 Red Hat Ceph Storage Object Gateway 指南中的使用 CephOrchestrator 部署多站点 Ceph 对象网关部分。

5.5.1. 删除存档区中的对象

您可以使用 S3 生命周期策略扩展来删除 < ArchiveZone> 元素中的对象。



重要

归档区对象只能使用 过期 生命周期策略规则删除。

- 如果任何 <Rule> 部分包含一个 <ArchiveZone> 元素，则该规则在归档区中执行，并且只能执行在归档区中运行的规则。

- 标记为 <ArchiveZone> 的规则不会在非存档区域中执行。

生命周期策略中的规则决定了要删除的对象。有关生命周期创建和管理的更多信息，请参阅 Bucket 生命周期。

先决条件

- 一个正在运行的 Red Hat Ceph Storage 集群。

- Ceph 监控节点的根级别访问权限.

- 安装 Ceph 对象网关软件.

流程

1. 设置 <ArchiveZone> 生命周期策略规则。有关创建生命周期策略的更多信息，请参阅 Red Hat Ceph Storage Object Gateway 指南中的创建生命周期管理策略部分。 https://access.redhat.com/documentation/zh-cn/red_hat_ceph_storage/7/html-single/object_gateway_guide/#creating-a-lifecycle-management-policy

   示例

```
<?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. ***可选：查看特定生命周期策略是否包含归档区规则。***

语法

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

示例

```
[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)

归档区规则。这是带有归档区规则的生命周期策略示例。

3.

如果删除了 Ceph 对象网关用户，则无法访问由该用户拥有的存档站点上的 bucket。将这些 bucket 链接到另一个 Ceph 对象网关用户，以访问数据。

语法

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

示例

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

其他资源

- 如需了解更多详细信息，请参阅 Red Hat Ceph Storage 对象网关指南中的 Bucket 生命周期 部分。

- 如需了解更多详细信息，请参阅 Red Hat Ceph Storage Developer Guide 中的 S3 存储桶生命周期 部分。

5.6. 故障转移和灾难恢复

如果主区失败，请切换到 second zone 进行灾难恢复。

先决条件

- 一个正在运行的 Red Hat Ceph Storage 集群。
- Ceph 监控节点的根级别访问权限.
- 安装 Ceph 对象网关软件.

流程

1. 将 second 区域设为主要和默认区域。例如：

语法

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

默认情况下，Ceph 对象网关在主动-主动配置中运行。如果集群配置为以主动-被动配置运行，则 second zone 是只读区域。删除 --read-only 状态，以允许区域接收写入操作。例如：

语法

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

2. **更新周期以使更改生效：**

示例

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

3. **重新启动 Ceph 对象网关。**

注意

使用 ceph orch ps 命令的输出，在 NAME 列下获取 SERVICE_TYPE。ID 信息。

a. **在存储集群中的单个节点上重启 Ceph 对象网关：**

语法

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

示例

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

b. **在存储集群的所有节点上重启 Ceph 对象网关：**

语法

```
ceph orch restart SERVICE_TYPE
```

示例

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

如果前一个主区域恢复，请恢复操作。

1. **在恢复的区中，从当前主区拉取域：**

语法

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

2. **使恢复的区域成为主要和默认区：**

语法

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

3. ***更新周期以使更改生效：***

示例

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

4. ***在恢复的区域中重启 Ceph 对象网关：***

语法

```
ceph orch restart SERVICE_TYPE
```

示例

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

5. ***如果 second zone 需要是一个只读配置，请更新 second zone：***

语法

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

6. **更新周期以使更改生效：**

示例

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

7. **在 second zone 中重启 Ceph 对象网关：**

语法

```
ceph orch restart SERVICE_TYPE
```

示例

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

5.7. 配置多个区域而无需复制

您可以配置多个不会相互复制的区域。例如，您可以为公司中的每个团队创建一个专用区。

先决条件

- 一个正在运行的 Red Hat Ceph Storage 集群。
- 安装 Ceph 对象网关软件.
- Ceph 对象网关节点的根级别访问权限.

流程

1. 创建新域：

语法

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

示例

```
[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. 创建新区组：

语法

语法

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

示例

```
[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. ***根据用例创建一个或多个区：***

语法

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

示例

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

4. **使用 zone group 的配置获取 JSON 文件：**

语法

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

示例

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

a. **打开 文件进行编辑，并将 log_meta、log_data 和 sync_from_all 字段设置为 false ：**

示例

```
{
    "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. ***使用更新的 JSON 文件设置 zone group：***

语法

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

示例

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

6. ***更新周期：***

示例

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

7. ***验证两个区是否已成功禁用：***

示例

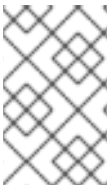
```
[root@ceph-ck-multi-pst82t-node5 ~]# radosgw-admin sync status

realm 1e513df5-279b-4558-9dd0-3e50af411740 (india)
zonegroup 09d320dd-d9f8-4fce-b951-8a59306a2d85 (south)
zone a88defd9-84f4-4a4e-8b21-7f3fbc190005 (ka)
current time 2024-03-15T05:01:18Z
zonegroup features enabled: compress-encrypted,resharding
metadata sync no sync (zone is master)
data sync source: 7a1ad335-9e09-403a-879c-d29cd81e9c4d (tn)
not syncing from zone
```

其它资源

- *Realms*
- *zone group*
- *zones*
- *安装指南*

5.8. 在同一存储集群中配置多个域

您可以在同一存储集群中配置多个域。这是多站点更高级的用例。在同一存储集群中配置多个域可让您使用本地域来处理本地 Ceph 对象网关客户端流量，以及复制到次要站点的数据的复制域。



注意

红帽建议每个域具有自己的 Ceph 对象网关。

先决条件

- *在存储集群中运行 Red Hat Ceph Storage 数据中心的两个.*
- *存储集群中每个数据中心的访问密钥和密钥。*
- *对所有 Ceph 对象网关节点的 root 级别访问。*
- *每个数据中心都有自己的本地域。它们共享两个站点上复制的域。*

流程

1. *在存储集群的第一个数据中心上创建一个本地域：*

语法

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

示例

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

2. **在第一个数据中心上创建一个本地 master zonegroup：**

语法

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

示例

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

3. **在第一个数据中心创建一个本地区：**

语法

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

示例

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

4. **提交周期：**

示例

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

5. **您可以使用适当的 realm 和 zone 部署 Ceph 对象网关守护进程，或更新配置数据库：**

- **使用放置规格部署 Ceph 对象网关：**

语法

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

示例

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

- ***更新 Ceph 配置数据库：***

语法

```
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
```

示例

```
[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. ***重新启动 Ceph 对象网关。***

注意

使用 ceph orch ps 命令的输出，在 NAME 列下获取 SERVICE_TYPE。ID 信息。

a. ***在存储集群中的单个节点上重启 Ceph 对象网关：***

语法

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

示例

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

b. ***在存储集群的所有节点上重启 Ceph 对象网关：***

语法

```
ceph orch restart SERVICE_TYPE
```

示例

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

7. ***在存储集群中的第二个数据中心上创建一个本地域：***

语法

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

示例

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

8. ***在第二个数据中心上创建一个本地 master zonegroup：***

语法

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

示例

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

9. ***在第二个数据中心创建一个本地区：***

语法

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

示例

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

10. **提交周期：**

示例

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

11. **您可以使用适当的 realm 和 zone 部署 Ceph 对象网关守护进程，或更新配置数据库：**

- **使用放置规格部署 Ceph 对象网关：**

语法

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

示例

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

- **更新 Ceph 配置数据库：**

语法

```
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
```

示例

```
[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. **重新启动 Ceph 对象网关。**

注意

使用 ceph orch ps 命令的输出，在 NAME 列下获取 SERVICE_TYPE。ID 信息。

a.

在存储集群中的单独节点上重启 Ceph 对象网关：

语法

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

示例

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

b.

在存储集群的所有节点上重启 Ceph 对象网关：

语法

```
ceph orch restart SERVICE_TYPE
```

示例

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

13.

在存储集群的第一个数据中心上创建一个复制的域：

语法

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

示例

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

使用 --default 标志在主站点上进行复制的域默认。

14. *为第一个数据中心创建一个 **master zonegroup**：*

语法

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

示例

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

15. ***在第一个数据中心上创建一个 master zone：***

语法

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

示例

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

16. ***创建同步用户，并将 system 用户添加到多站点的 master 区中：***

语法

```
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
```

示例

```
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=VpvQWcsfI9OPzUCpR4kynDLAbqa1OlKqRB6WEnH8
```

17. ***提交周期：***

示例

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

18. ***您可以使用适当的 realm 和 zone 部署 Ceph 对象网关守护进程，或更新配置数据库：***

- ***使用放置规格部署 Ceph 对象网关：***

语法

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

示例

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

- ***更新 Ceph 配置数据库：***

语法

```
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
```

示例

```
[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. ***重新启动 Ceph 对象网关。***

注意

使用 ceph orch ps 命令的输出，在 NAME 列下获取 SERVICE_TYPE。ID 信息。

a. ***在存储集群中的单独节点上重启 Ceph 对象网关：***

语法

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

示例

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

b. ***在存储集群的所有节点上重启 Ceph 对象网关：***

语法

```
ceph orch restart SERVICE_TYPE
```

示例

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

20. ***在第二个数据中心上拉取复制域：***

语法

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

示例

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

21. ***从第一个数据中心拉取 period：***

语法

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

示例

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

22. ***在第二个数据中心中创建 second zone：***

语法

```
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
```

示例

```
[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. **提交周期：**

示例

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

24. **您可以使用适当的 realm 和 zone 部署 Ceph 对象网关守护进程，或更新配置数据库：**

- **使用放置规格部署 Ceph 对象网关：**

语法

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

示例

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

- **更新 Ceph 配置数据库：**

语法

```
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
```

示例

```
[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. **重新启动 Ceph 对象网关。**

注意

使用 ceph orch ps 命令的输出，在 NAME 列下获取 SERVICE_TYPE。ID 信息。

a.

在存储集群中的单独节点上重启 Ceph 对象网关：

语法

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

示例

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

b.

在存储集群的所有节点上重启 Ceph 对象网关：

语法

```
ceph orch restart SERVICE_TYPE
```

示例

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

26.

以 root 用户身份登录第二个数据中心的端点。

27. *验证 **master 域上的同步状态：***

语法

```
radosgw-admin sync status
```

示例

```
[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. ***以 root 用户身份登录第一个数据中心的端点。***

29. *验证 **replication-synchronization 域的同步状态：***

语法

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

示例

```
[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. **要在本地站点中存储和访问数据，请为本地域创建用户：**

语法

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

示例

```
[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

默认情况下，用户在默认域下创建用户。要使用户访问本地域中数据的用户，radosgw-admin 命令需要 --rgw-realm 参数。

5.9. 使用多站点同步策略

作为存储管理员，您可以在存储桶级别上使用多站点同步策略来控制不同区域中 bucket 之间的数据移动。这些策略称为存储桶粒度同步策略。在以前的版本中，区中的所有存储桶都是对称的。这意味着每个区域都包含给定存储桶的镜像副本，并且存储桶副本在所有区域中都相同。同步过程假定存储桶同步源和存储桶同步目的地引用同一存储桶。

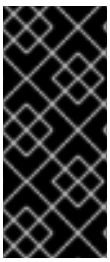
重要

bucket 同步策略只适用于数据，无论存储桶同步策略存在，元数据仅在多站点中的所有区域间同步。当存储桶同步策略处于 allowed 或 forbidden 时，创建、修改或删除的对象，它不会在策略生效时自动同步。运行 bucket sync run 命令以同步这些对象。

重要

如果在 zonegroup 级别定义了多个同步策略，则任何时间点上只能有一个策略处于 enabled 状态。如果需要，我们可以在策略间切换

同步策略替代旧的 zone group coarse 配置(sync_from*)。同步策略可以在 zone group 级别上配置。如果配置了该配置，它将替换 zone group 级别的旧式配置，但也可在 bucket 级别上进行配置。



重要

bucket 同步策略适用于归档区域。归档区的移动不是双向的，其中所有对象都可以从活动区域移动到存档区域。但是，您无法将对象从存档区域移到 active zone，因为 archive 区域是只读的。

zone group 的存储桶同步策略示例

```
[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"
            },
            ...
        }
    ],
    ...
}
```

先决条件

- **正在运行的 Red Hat Ceph Storage 集群。**
- **Ceph 监控节点的根级别访问权限.**

- 安装 Ceph 对象网关软件.

5.9.1. 多站点同步策略组状态

在同步策略中，可以定义可以包含 data-flow 配置列表的多个组，以及管道配置列表。data-flow 定义不同区域之间的数据流。它可以定义对称数据流，其中有多个区域同步数据，并且可以定义方向数据流，其中的数据从一个区域移动到另一个区域。

管道定义可以使用这些数据流的实际存储桶，以及与它关联的属性，如源对象前缀。

同步策略组可能处于 3 个状态：

- 允许并启用启用。
- 允许允许 criu-wagonsync。
- 不允许在此组定义 禁止 ProductShortName-wagonsync。

复制区域时，您可以使用同步策略禁用特定存储桶的复制。以下是需要遵循解决策略冲突的语义：

Zonegroup	Bucket	结果
enabled	enabled	enabled
enabled	allowed	enabled
enabled	禁止	disabled
allowed	enabled	enabled
allowed	allowed	disabled
allowed	禁止	disabled
禁止	enabled	disabled

Zonegroup	Bucket	结果
禁止	allowed	disabled
禁止	禁止	disabled

对于被设置为反映任何同步对(SOURCE_ZONE、SOURCE_BUCKET)、(DESTINATION_ZONE,DESTINATION_BUCKET)的多个组策略，按以下顺序应用以下规则：

- *即使 禁止 一个同步策略，同步也会 禁用。*
- *应至少 启用 一个策略，以便允许同步。*

这个组中的同步状态可能会覆盖其他组。

策略可以在 bucket 级别上定义。bucket 级别同步策略继承 zonegroup 策略的数据流，并且只能定义 zonegroup 允许的子集。

通配符区域，策略中的通配符 bucket 参数定义所有相关区域或所有相关存储桶。在 bucket 策略的上下文中，这意味着当前的 bucket 实例。灾难恢复配置，其中整个区被镜像(mirror)不需要在存储桶上配置任何内容。但是，对于精细的存储桶同步，最好在 zonegroup 级别允许(status=allowed)将它们配置为同步管道（例如，使用通配符）。但是，仅在存储桶级别(status=enabled)中启用特定的同步。如果需要，存储桶级别的策略可以将数据移动限制到特定的相关区域。

重要

对 zonegroup 策略的任何更改都需要应用到 zonegroup master zone，并且需要周期更新和提交。对 bucket 策略的更改需要应用到 zonegroup master zone。Ceph 对象网关动态处理这些更改。

S3 bucket 复制 API

S3 bucket 复制 API 已被实施，允许用户在不同 bucket 之间创建复制规则。请注意，虽然 AWS 复制功能允许在同一区内复制存储桶，但 Ceph 对象网关目前不允许它。但是，Ceph 对象网关 API 也添加了一个 Zone 数组，允许用户选择要同步特定 bucket 的区域。

其它资源

- 如需了解更多详细信息，请参阅 S3 存储桶复制 API。

5.9.2. 检索当前策略

您可以使用 get 命令检索当前 zonegroup 同步策略或特定的存储桶策略。

先决条件

- 正在运行的 Red Hat Ceph Storage 集群。
- root 或 sudo 访问权限。
- 已安装 Ceph 对象网关。

流程

- 检索当前的 zonegroup 同步策略或存储桶策略。要检索特定的存储桶策略，请使用 --bucket 选项：

  语法

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

  示例

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

5.9.3. 创建同步策略组

您可以为当前 zone group 或特定存储桶创建同步策略组。

当为存储桶粒度复制创建同步策略时，从 禁止 变为 启用的 同步策略组，可能需要手动更新才能完成同步过程。

例如，如果 禁止 其策略将任何数据写入 bucket1，则在策略更改为 启用 后，数据可能无法在区域间正确同步。要正确同步更改，请在同步策略上运行 bucket sync run 命令。如果策略 禁止 时 bucket 重新划分，则还需要这一步。在这种情况下，在启用策略后，还必须使用 bucket sync run 命令。

先决条件

- 正在运行的 Red Hat Ceph Storage 集群。
- root 或 sudo 访问权限。
- 已安装 Ceph 对象网关。
- 在为归档区创建时，请确保在同步策略组之前创建存档区域。

流程

1. 创建同步策略组或存储桶策略。要创建存储桶策略，请使用 --bucket 选项：

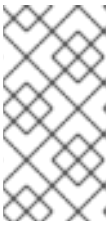
语法

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

示例

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

2. 可选：手动完成存储桶粒度复制的同步过程。



注意

如果将 用作带有存储桶粒度复制的归档区域的一部分时，此步骤是必需的，如果策略 禁止 了数据，或者存储桶被重新划分。

语法

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

示例

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

其它资源

有关配置存档区域和存储桶粒度复制的更多信息，请参阅配置归档区。

5.9.4. 修改同步策略组

您可以修改当前 zone group 或特定存储桶的现有同步策略组。

当为从 禁止 变为 启用 的同步策略组修改存储桶粒度复制的同步策略时，可能需要手动更新才能完成

同步过程。

例如，如果 禁止 其策略将任何数据写入 bucket1，则在策略更改为 启用 后，数据可能无法在区域间正确同步。要正确同步更改，请在同步策略上运行 bucket sync run 命令。如果策略 禁止 时 bucket 重新划分，则还需要这一步。在这种情况下，在启用策略后，还必须使用 bucket sync run 命令。

先决条件

- 正在运行的 Red Hat Ceph Storage 集群。
- root 或 sudo 访问权限。
- 已安装 Ceph 对象网关。
- 修改存档区域时，请确保在同步策略组之前创建存档区域。

流程

1. 修改同步策略组或存储桶策略。要修改存储桶策略，请使用 --bucket 选项。

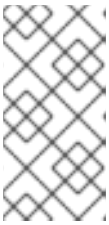
语法

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

示例

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

2. 可选：手动完成存储桶粒度复制的同步过程。



注意

如果将 用作带有存储桶粒度复制的归档区域的一部分时，此步骤是必需的，如果策略 禁止 了数据，或者存储桶被重新划分。

语法

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

示例

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

其它资源

有关配置存档区域和存储桶粒度复制的更多信息，请参阅配置归档区。

5.9.5. 获取同步策略组

您可以使用 **group get** 命令按组 **ID** 显示当前的同步策略组，或者显示特定的存储桶策略策略。

如果没有提供 **--bucket** 选项，则在 **zonegroup** 级别中创建的组会被检索，而不是 **bucket** 级别上的组。

先决条件

- 正在运行的 Red Hat Ceph Storage 集群。
- root 或 sudo 访问权限。
- 已安装 Ceph 对象网关。

流程

- 显示当前的同步策略组或存储桶策略。要显示特定的存储桶策略，请使用 --bucket 选项：

  语法

```
radosgw-admin sync group get --bucket=BUCKET_NAME --group-id=GROUP_ID
```

  示例

```
[ceph: root@host01 /]# radosgw-admin sync group get --group-id=mygroup
```

5.9.6. 删除同步策略组

您可以使用 group remove 命令按组 ID 删除当前的同步策略组，或者删除特定的存储桶策略策略。

先决条件

- 正在运行的 Red Hat Ceph Storage 集群。
- root 或 sudo 访问权限。

- 已安装 Ceph 对象网关。

流程

- 删除当前的同步策略组或存储桶策略。要删除特定的存储桶策略，请使用 **--bucket** 选项：

语法

```
radosgw-admin sync group remove --bucket=BUCKET_NAME --group-id=GROUP_ID
```

示例

```
[ceph: root@host01 /]# radosgw-admin sync group remove --group-id=mygroup
```

5.9.7. 创建同步流

您可以为同步策略组或特定存储桶创建两种不同类型的流：

- 方向性同步流
- 对称同步流

group flow create 命令创建一个同步流。如果您为已具有同步流的同步策略组或存储桶发出 **group flow create** 命令，该命令将覆盖同步流的现有设置，并应用您指定的设置。

选项	描述	必填/选填

选项	描述	必填/选填
--bucket	需要配置同步策略的存储桶的名称。仅在 bucket 级别同步策略中使用。	选填
--group-id	同步组的 ID。	必填
--flow-id	流的 ID。	必填
--flow-type	同步策略组或特定存储桶的流类型 - 方向或对称。	必填
--source-zone	指定应发生同步的源区。将数据发送到同步组的区域。如果同步组的流类型为方向，则需要此项。	选填
--dest-zone	指定应发生同步的目标区。从同步组接收数据的区域。如果同步组的流类型为方向，则需要此项。	选填
--zones	同步组一部分的区域。区域提及是发送者和接收器区域。指定以 "," 分隔的区域。如果同步组的流类型是 symmetrical，则需要此项。	选填

先决条件

- ***一个正在运行的 Red Hat Ceph Storage 集群。***
- ***root 或 sudo 访问权限。***
- ***已安装 Ceph 对象网关。***

流程

1. ***创建或更新方向同步流。要为特定存储桶创建或更新同步流，请使用 --bucket 选项。***

   语法

```
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. 创建或更新对称同步流。要为对称流类型指定多个区域，请在 **--zones** 选项中使用逗号分隔的列表。

语法

```
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 是需要添加到流中的所有区的逗号分隔列表。

5.9.8. 删除同步流和区域

group flow remove 命令从同步策略组或存储桶中删除同步流或区域。

对于使用方向流的同步策略组或 **bucket**，**group flow remove** 命令会删除流。对于使用对称流的同步策略组或 **bucket**，您可以使用 **group flow remove** 命令从流中删除指定区域，或者移除流。

先决条件

- 正在运行的 **Red Hat Ceph Storage** 集群。
- **root** 或 **sudo** 访问权限。
- 已安装 **Ceph** 对象网关。

流程

1. 删除方向同步流。要删除特定存储桶的方向同步流，请使用 **--bucket** 选项。

语法

```
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. 从对称同步流中删除特定区域。要从对称流中删除多个区域，请对 **--zones** 选项使用逗号分隔的列表。

语法

```
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. 移除对称同步流。要删除 **zonegroup** 级别的同步流，请删除 **--bucket** 选项。

语法

```
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. 创建或修改同步组管道

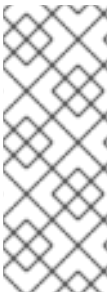
作为存储管理员，您可以定义管道来指定哪些存储桶可以使用您配置的数据流，以及与这些数据流关联的属性。

sync group pipe create 命令允许您创建管道，它们是特定存储桶或 bucket 组或特定区域组之间的自定义同步组数据流。

这个命令使用以下选项：

选项	描述	必填/选填
--bucket	需要配置同步策略的存储桶的名称。仅在 bucket 级别同步策略中使用。	选填
--group-id	同步组的 ID	必填
--pipe-id	管道 ID	必填
--source-zones	将数据发送到同步组的区域。使用单引号(')作为值。使用逗号分隔多个区域。对于与数据流规则匹配的所有区域，使用通配符 *。	必填
--source-bucket	将数据发送到同步组的 bucket 或 bucket。如果没有提及存储桶名称，则会将 * （通配符）用作默认值。在 bucket 级别，源 bucket 将是创建在 zonegroup-level 的同步组以及 zonegroup-level 的存储桶，源存储桶将是所有存储桶。	选填
--source-bucket-id	源存储桶的 ID。	选填
--dest-zones	接收同步数据的区域或区域。使用单引号(')作为值。使用逗号分隔多个区域。对于与数据流规则匹配的所有区域，使用通配符 *。	必填
--dest-bucket	接收同步数据的 bucket。如果没有提及存储桶名称，则会将 * （通配符）用作默认值。在 bucket 级别，目标存储桶将是为其创建同步组的存储桶，在 zonegroup-level 中，目标存储桶将是所有存储桶	选填
--dest-bucket-id	目标 bucket 的 ID。	选填
--prefix	bucket 前缀。使用通配符 * 过滤源对象。	选填

选项	描述	必填/选填
--prefix-rm	不要使用 bucket 前缀进行过滤。	选填
--tags-add	键=值对的逗号分隔列表。	选填
--tags-rm	删除一个或多个标签的 key=value 对。	选填
--dest-owner	来自源的对象的目的地所有者。	选填
--storage-class	来自源的对象的目的地存储类。	选填
--mode	将 **system** 用于系统模式或 **user** 作为用户模式。	选填
--uid	用于用户模式的权限验证。指定将发出同步操作的用户 ID。	选填



注意

如果要在 zonegroup 级别上为特定存储桶启用/禁用同步，请将 zonegroup 级别同步策略设置为 enable/disable，并为带有 --source-bucket 和 --dest-bucket 的每个存储桶创建一个管道，并使用相同的存储桶名称或 bucket-id、i.e、--source-bucket-id 和 --dest-bucket-id。

先决条件

- ***一个正在运行的 Red Hat Ceph Storage 集群。***
- ***root 或 sudo 访问权限。***
- ***已安装 Ceph 对象网关。***

流程

- ***创建同步组管道。create 命令还用于通过使用相关选项创建同步组管道来更新命令。***

语法

```
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. 修改或删除同步组管道

作为存储管理员，您可以使用 **sync group pipe modify** 命令或 **sync group pipe remove** 命令，通过删除某些选项来修改同步组管道。您还可以使用 **sync group pipe remove** 命令完全删除区域、存储桶或同步组管道。

先决条件

- 一个正在运行的 **Red Hat Ceph Storage** 集群。
- **root** 或 **sudo** 访问权限。
- 已安装 **Ceph** 对象网关。

流程

- 使用 **modify** 参数修改同步组管道选项。

语法

```
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
```

注意

确保将区域放在单引号(')中。源存储桶不需要引号。

示例

```
[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
```

- **使用 remove 参数修改同步组管道选项。**

语法

```
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
```

示例

```
[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
```

- **删除同步组管道。**

语法

```
radosgw-admin sync group pipe remove --bucket=BUCKET_NAME --group-id=GROUP_ID --pipe-id=PIPE_ID
```

示例

```
[root@host01 ~]# radosgw-admin sync group pipe remove -bucket-name=mybuck --group-id=zonegroup --pipe-id=pipe
```

5.9.11. 获取同步操作的信息

sync info 命令可让您获取有关同步策略所定义的预期同步源和目标的信息。

当您为存储桶创建同步策略时，该策略定义数据如何从该存储桶移动到不同区域的不同 **bucket**。创建策略也会创建一个 **bucket** 依赖项列表，每当该 **bucket** 与其他存储桶同步时，它们就可用作提示。请注意，存储桶可以引用另一个存储桶，而无需实际同步，因为同步取决于数据流是否允许同步进行。

--bucket 和 **effective-zone-name** 参数是可选的。如果您调用 **sync info** 命令但不指定任何选项，则对象网关会返回所有区域中同步策略定义的所有同步操作。

先决条件

- 正在运行的 **Red Hat Ceph Storage** 集群。
- **root** 或 **sudo** 访问权限。
- 已安装 **Ceph** 对象网关。

- 定义了组同步策略。

流程

- 获取存储桶的同步操作信息：

  语法

```
radosgw-admin sync info --bucket=BUCKET_NAME --effective-zone-name=ZONE_NAME
```

- 在 zonegroup 级别获取有关同步操作的信息：

  语法

```
radosgw-admin sync info
```

5.10. BUCKET 粒度同步策略

现在支持以下功能：

- greenfield 部署 ：此发行版本支持新的多站点部署。要设置存储桶粒度同步复制，必须至少配置一个新的 zonegroup/zone。

- brownfield 部署 ：将 Ceph 对象网关多站点复制配置迁移到新功能的 Ceph 对象网关 bucket 粒度同步策略复制。

注意

在升级过程中，确保存储集群中的所有节点都位于同一 schema。

- 数据流 - 方向、对称 ：可以配置单向和双向/符号链接复制。

重要

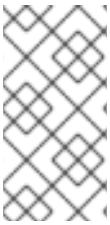
在这个发行版本中，不支持以下功能：

- 源过滤器
- Storage class
- 目标所有者转换
- 用户模式

当存储桶或 zonegroup 的同步策略时，从 disabled 状态移到 enabled 状态，观察以下行为更改：

正常场景 ：

- zonegroup 级别：当同步策略被禁用后，当它被启用时会写入数据，无需额外的步骤。
- bucket 级别：当 禁用 同步策略时写入的数据 在启用 策略时不会捕获。在这种情况下，可以应用以下两个临时解决方案之一：
  - 向存储桶写入新数据会重新同步旧数据。
  - 执行 bucket sync run 命令同步所有旧数据。



注意

当您要从同步策略切换到旧策略时，您需要首先运行 sync init 命令，后跟 radosgw-admin bucket sync run 命令来同步所有对象。

重新定义场景 ：

- zonegroup 级别： 禁用 策略时发生的任何重新划分，同步会在策略再次启用后卡住。此时，新对象也不会同步。作为临时解决方案运行 bucket sync run 命令。
- bucket 级别：如果在策略 被禁用时 重新分片任何存储桶，则同步会在策略再次启用后卡住。此时，新对象也不会同步。作为临时解决方案运行 bucket sync run 命令。

注意

当为 zonegroup 设置为 enabled 且策略被设置为 enabled 或 allowed bucket 时，管道配置会从 zonegroup 级别而不是 bucket 级别生效。这是一个已知问题 BZ#2240719。

5.10.1. 为 zonegroups 设置双向策略

使用新的同步策略引擎创建 zonegroup 同步策略。对 zonegroup 同步策略的任何更改都需要一个周期更新和提交。

在以下示例中，创建一个组策略，并定义一个数据流，用于将数据从一个区域移动到另一个区域。配置 zonegroups 的管道，以定义可使用此数据流的存储桶。以下示例中的系统包括 3 个区域： us-east (master zone)、us-west 和 us-west-2。

先决条件

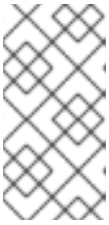
- 一个正在运行的 Red Hat Ceph Storage 集群。
- 已安装 Ceph 对象网关。

流程

1. 创建新的同步组，并将状态设置为 允许。

示例

```
[ceph: root@host01 /]# radosgw-admin sync group create --group-id=group1 --
status=allowed
```



注意

在创建完全配置的 **zonegroup** 复制策略前，建议将 **--status** 设置为 **allowed**，以防止复制启动。

2. 为新创建的组创建一个流策略，并将 **--flow-type** 设置为 **symmetrical** 来启用双向复制。

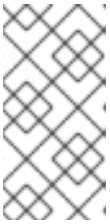
示例

```
[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. 创建名为 **pipe** 的新管道。

示例

```
[ceph: root@host01 /]# radosgw-admin sync group pipe create --group-id=group1 \
                        --pipe-id=pipe1 --source-zones='*' \
                        --source-bucket='*' --dest-zones='*' \
                        --dest-bucket='*'
```



注意

使用 * 通配符用于区，使其包含之前流策略中设置的所有区域，而 * 用于存储桶在区域中复制所有现有存储桶。

4. 配置存储桶同步策略后，将 **--status** 设置为 **enabled**。

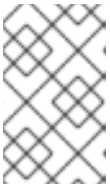
示例

```
[ceph: root@host01 /]# radosgw-admin sync group modify --group-id=group1 --
status=enabled
```

5. 更新并提交新周期。

示例

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



注意

对于 **zonegroup** 策略，更新和提交周期是强制的。

6. 可选：检查特定存储桶的同步源和目的地。区域 **us-east** 和 **us-west** 中的所有存储桶都以双向方式复制。

示例

```
[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"
            },
           ...
        }
    ],
    ...
}
```

以上输出中的 id 字段反映了生成该条目的管道规则。单个规则可以生成多个同步条目，如以下示例中所示。

5.10.2. 为 zonegroups 设置方向策略

使用同步策略引擎以方向方式设置 zone group 的策略。

在以下示例中，创建一个组策略，并配置数据流，以便将数据流从一个区域移动到另一个区域。另外，为 zonegroups 配置管道，以定义可使用此数据流的存储桶。以下示例中的系统包括 3 个区域： us-

east （主区域）、us-west (secondary zone)和 us-west-2 （备份区域）。在这里，us-west-2 是 us-west 的副本，但数据不会从其中复制。

先决条件

- *一个正在运行的 Red Hat Ceph Storage 集群。*
- *已安装 Ceph 对象网关。*

流程

1. *在 primary zone 上，创建一个新的同步组，并将状态设置为 允许。*

   语法

   ```
   radosgw-admin sync group create --group-id=GROUP_ID --status=allowed
   ```

   示例

   ```
   [ceph: root@host01 /]# radosgw-admin sync group create --group-id=group1 --status=allowed
   ```

   注意

   在创建完全配置的 zonegroup 复制策略前，建议将 --status 设置为 allowed，以防止复制启动。

2. *创建流。*

   语法

```
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
```

示例

```
[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. **创建管道。**

语法

```
radosgw-admin sync group pipe create --group-id=GROUP_ID --pipe-id=PIPE_ID  --source-zones='SOURCE_ZONE_NAME' --dest-zones='DESTINATION_ZONE_NAME'
```

示例

```
[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. **更新并提交新周期。**

示例

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

注意

对于 **zonegroup** 策略，更新和提交周期是强制的。

5. 使用两个站点的同步信息，验证 **zonegroup** 的源和目的地。

语法

```
radosgw-admin sync info
```

5.10.3. 为存储桶设置方向策略

使用同步策略引擎以说明方式设置 **bucket** 的策略。

在以下示例中，创建一个组策略，并配置数据流，以便将数据流从一个区域移动到另一个区域。另外，为 **zonegroups** 配置管道，以定义可使用此数据流的存储桶。以下示例中的系统包括 **3** 个区域： **us-east** （主区域）、**us-west (secondary zone)**和 **us-west-2** （备份区域）。在这里，**us-west-2** 是 **us-west** 的副本，但数据不会从其中复制。

为 **zonegroups** 和 **bucket** 设置方向策略之间的区别是您需要指定 **--bucket** 选项。

先决条件

- 一个正在运行的 **Red Hat Ceph Storage** 集群。

- 已安装 Ceph 对象网关。

流程

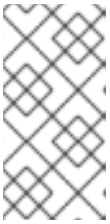
1. 在 primary zone 上，创建一个新的同步组，并将状态设置为 允许。

语法

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

示例

```
[ceph: root@host01 /]# radosgw-admin sync group create --group-id=group1 --
status=allowed --bucket=buck
```



注意

在创建完全配置的 zonegroup 复制策略前，建议将 --status 设置为 allowed，以防止复制启动。

2. 创建流。

语法

```
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
```

示例

```
[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. *创建管道。*

语法

```
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'
```

示例

```
[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'
```

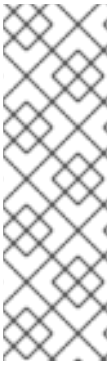
4. *使用两个站点的同步信息，验证* ***zonegroup*** *的源和目的地。*

语法

```
radosgw-admin sync info --bucket-name=BUCKET_NAME
```

5.10.4. 为存储桶设置双向策略

bucket 级别策略的数据流继承自 zonegroup 策略。bucket 级别策略不需要更改数据流和管道，因为 bucket 级别策略流和管道只是 zonegroup 策略中定义的流的子集。



注意

- 存储 同级别策略可以启用还没有启用的管道，但在 zonegroup 策略中被禁止的除外。
- bucket 级别策略不需要周期更新。

先决条件

- 一个正在运行的 Red Hat Ceph Storage 集群。
- 已安装 Ceph 对象网关。
- 已创建一个同步流。

流程

1. 将 zonegroup 策略 --status 设置为 allowed per-bucket 复制。

示例

```
[ceph: root@host01 /]# radosgw-admin sync group modify --group-id=group1 --
status=allowed
```

2. 修改 zonegroup 策略后更新周期。

示例

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

3. 为要同步到的存储桶创建同步组，并将 --status 设置为 enabled。

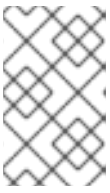
示例

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

4. 为上一步中创建的组创建管道。流继承自 zonegroup 级别策略，其中数据流是对称的。

示例

```
[ceph: root@host01 /]# radosgw-admin sync group pipe create --bucket=buck \
                        --group-id=buck-default --pipe-id=pipe1 \
                        --source-zones='*' --dest-zones='*'
```



注意

使用通配符 * 为存储桶复制指定源和目标区域。

5. 可选：要检索有关同步策略定义的预期存储桶同步源和目标的信息，请使用 --bucket 标志运

*行 **radosgw-admin bucket sync info** 命令。*

示例

```
[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. *可选：要检索同步策略定义的预期同步源和目标的信息，请使用 **--bucket** 标志运行 **radosgw-admin sync info** 命令。*

示例

```
[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. 在存储桶间同步

您可以在区域间同步源和目标存储桶之间的数据，但不能在同一区中同步。请注意，内部，数据仍然从目标区的源中拉取。

通配符存储桶名称表示当前的存储桶位于 bucket 同步策略上下文中。

bucket 之间有两种同步：

1. 从存储桶同步 - 您需要指定源存储桶。
2. 同步到存储桶 - 您需要指定目标存储桶。

先决条件

- 一个正在运行的 Red Hat Ceph Storage 集群。

- 已安装 Ceph 对象网关。

从不同的存储桶同步

1. 创建同步组，从另一个区域的存储桶中提取数据。

语法

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

示例

```
[ceph: root@host01 /]# radosgw-admin sync group create --bucket=buck4 --group-id=buck4-default --status=enabled
```

2. 拉取镜像。

语法

```
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
```

示例

```
[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='*'
```

在本例中，您可以看到源存储桶为 ***buck5。***

3. *可选：从特定区域中的存储桶同步。*

示例

```
[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. *检查同步状态。*

语法

```
radosgw-admin sync info --bucket-name=BUCKET_NAME
```

示例

```
[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": []
    }
```

*请注意，有 **resolved-hints**，这意味着存储桶 **buck5** 发现了它间接的 **buck4** 同步，而不是其自身的策略。**buck5** 本身的策略为空。*

同步到不同的存储桶

1. *创建同步组。*

语法

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

示例

```
[ceph: root@host01 /]# radosgw-admin sync group create --bucket=buck6 \
                      --group-id=buck6-default --status=enabled
```

2. **推送数据。**

语法

```
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
```

示例

```
[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
```

在本例中，您可以看到目标存储桶为 buck5。

3. *可选：与特定区中的存储桶同步。*

示例

```
[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. *检查同步状态。*

语法

```
radosgw-admin sync info --bucket-name=BUCKET_NAME
```

示例

```
[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. 过滤对象

使用前缀和标签过滤存储桶中的对象。您还可以在 zonegroup 级别策略上设置对象过滤器。如果使用 --bucket 选项，则在存储桶的存储桶级别上设置它。

在以下示例中，从一个区的 buck1 bucket 同步到另一个区域中的 buck1 存储桶，带有以 foo/ 前缀的对象。同样，您可以过滤具有 color=blue 等标签的对象。前缀和标签可以合并，其中对象需要同时同步。priority 参数也可以传递，它用于确定是否有多个匹配的不同规则。此配置决定了要使用的规则。

注意

1. *如果同步策略中的标签有多个标签，同时同步对象，它会同步与至少一个标签匹配的对象，键值对。对象可能与所有标签集不匹配。*

2. *如果设置了 prefix 和 tag，则要将对象同步到另一个区域，对象必须具有前缀，任何一个标签应匹配。只有这样，它会相互同步。*

先决条件

- 至少两个正在运行的 **Red Hat Ceph Storage** 集群。
- 已安装 **Ceph** 对象网关。
- **bucket** 已创建。

流程

1. 创建新的同步组。如果要在存储桶级别创建，请使用 **--bucket** 选项。

语法

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

示例

```
[ceph: root@host01 /]# radosgw-admin sync group create --bucket=buck1 --group-id=buck8-default --status=enabled
```

2. 对象与标签匹配的 **bucket** 之间的同步。流继承自 **zonegroup** 级别策略，其中数据流是对称的。

语法

```
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'
```

示例

```
[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. 在对象与前缀匹配的 **bucket** 之间进行同步。流继承自 **zonegroup** 级别策略，其中数据流是对称的。

语法

```
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'
```

示例

```
[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. 检查更新的同步。

语法

```
radosgw-admin sync info --bucket=BUCKET_NAME
```

示例

```
[ceph: root@host01 /]# radosgw-admin sync info --bucket=buck1
```

注意

在示例中，只有两个不同的目的地，没有源反映，每个源进行配置。当同步过程发生时，它会为其同步的每个对象选择相关规则。

5.10.7. 禁用存储桶之间的策略

您可以禁用存储桶之间的策略，以及与 禁止 或允许 的状态同步信息。

有关可用于 zonegroup 和 bucket 级别同步策略的不同组合，请参阅多站点同步策略 状态。

在某些情况下，若要中断两个存储桶之间的复制，您可以将存储桶的组策略设置为 禁止。如果任何存储桶没有发生同步，您也可以在 zonegroup 级别禁用策略。

注意

您还可以使用 radosgw-admin sync group create 命令在禁用状态下创建一个同步策略，并具有 禁止 状态。

先决条件

- 一个正在运行的 Red Hat Ceph Storage 集群。

- 已安装 Ceph 对象网关。

流程

1. 运行 sync group modify 命令，将允许 的状态更改为 禁止。

示例

```
[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"
        }
    ]
}
```

在本例中，存储桶 buck 的复制在区域 us-east 和 us-west 之间中断。

注意

不需要在周期内更新和提交，因为这是存储桶同步策略。

2. 可选：运行 sync info 命令检查 存储桶 buck 的同步状态。

示例

```
[ceph: root@host01 /]# radosgw-admin sync info --bucket buck
{
    "sources": [],
    "dests": [],
    "hints": {
        "sources": [],
        "dests": []
    },
    "resolved-hints-1": {
        "sources": [],
        "dests": []
    },
    "resolved-hints": {
        "sources": [],
        "dests": []
    }
}
```

注意

因为复制中断，所以没有源和目标目标。

5.11. 多站点 CEPH 对象网关命令行使用

作为存储管理员，您可以更好地了解如何在多站点环境中使用 Ceph 对象网关。您可以在多站点环境中了解如何更好地管理域、区域组和区域。

先决条件

- 正在运行的 Red Hat Ceph Storage
- 部署 Ceph 对象网关软件。
- 访问 Ceph 对象网关节点或容器。

5.11.1. Realms

realm 代表一个全局唯一的命名空间，它由一个或多个 zone group 组成，包含一个或多个 zone，以及包含 bucket 的区域，后者又包含对象。域允许 Ceph 对象网关在同一硬件上支持多个命名空间及其配置。

域中包含句点的概念。每个 period 代表 zone group 和 zone 配置的状态。每次您更改一个 zonegroup 或 zone zone 时，更新 period 并提交。

红帽建议为新集群创建域（realms）。

5.11.1.1. 创建一个域

要创建域，请发出 realm create 命令并指定 realm 名称。如果 realm 是默认值，指定 --default。

语法

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

示例

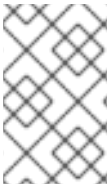
```
[ceph: root@host01 /]# radosgw-admin realm create --rgw-realm=test_realm --default
```

通过指定 **--default**，域将通过每个 **radosgw-admin** 调用隐式调用，除非明确提供了 **--rgw-realm** 和 **realm** 名称。

5.11.1.2. 将 Realm 设置为默认值

realm 列表中的一个域应为默认域。可能只有一个默认域。如果只有一个域，但没有在创建时指定为默认域，则使其成为默认域。另外，要更改哪个域是默认域，请运行以下命令：

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



注意

当 **realm** 为 **default** 时，命令行假设 **--rgw-realm=REALM_NAME** 作为参数。

5.11.1.3. 删除 Realm

要删除域，请运行 **realm delete** 命令并指定 **realm** 名称。

语法

```
radosgw-admin realm delete --rgw-realm=REALM_NAME
```

示例

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

5.11.1.4. 获取域

要获取域，请运行 **realm get** 命令并指定 **realm** 名称。

语法

```
radosgw-admin realm get --rgw-realm=REALM_NAME
```

示例

```
[ceph: root@host01 /]# radosgw-admin realm get --rgw-realm=test_realm >filename.json
```

CLI 将回显具有 **realm** 属性的 **JSON** 对象。

```
{
    "id": "0a68d52e-a19c-4e8e-b012-a8f831cb3ebc",
    "name": "test_realm",
    "current_period": "b0c5bbef-4337-4edd-8184-5aeab2ec413b",
    "epoch": 1
}
```

使用 **>** 和输出文件名将 **JSON** 对象输出到文件中。

5.11.1.5. 设置域

要设置域，请运行 **realm set** 命令，使用输入文件名指定 **realm** 名称和 **--infile=**。

语法

```
radosgw-admin realm set --rgw-realm=REALM_NAME --infile=IN_FILENAME
```

示例

```
[ceph: root@host01 /]# radosgw-admin realm set --rgw-realm=test_realm --infile=filename.json
```

5.11.1.6. 列出域

要列出 **realm**，请运行 **realm list** 命令：

示例

```
[ceph: root@host01 /]# radosgw-admin realm list
```

5.11.1.7. 列出 Realm Periods

要列出 **realm period**，请运行 **realm list-periods** 命令。

示例

```
[ceph: root@host01 /]# radosgw-admin realm list-periods
```

5.11.1.8. 拉取域

要将包含 master zone group 和 master zone 的节点的域拉取到包含 second zone group 或 zone 的节点，请在将接收 realm 配置的节点上运行 realm pull 命令。

语法

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

5.11.1.9. 重命名域

realm 不是该 period 的一部分。因此，仅在本地应用重命名域，且不会通过 realm pull 来拉取。重命名具有多个区域的域时，在每个区域上运行这个命令。要重命名域，请运行以下命令：

语法

```
radosgw-admin realm rename --rgw-realm=REALM_NAME --realm-new-
name=NEW_REALM_NAME
```

注意

不要使用 realm set 来更改 name 参数。这仅更改内部名称。指定 --rgw-realm 仍然会使用旧的域名。

5.11.2. zone group

Ceph 对象网关利用 zone group 的概念来支持多站点部署和全局命名空间。zone group 以前称为地区，它定义一个或多个 zone 中一个或多个 Ceph 对象网关实例的地理位置。

配置 zone group 与典型的配置过程不同，因为并非所有设置都最终在 Ceph 配置文件中。您可以列出 zone group、获取 zone group 配置并设置 zone group 配置。

注意

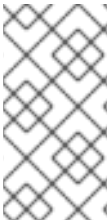
radosgw-admin zonegroup 操作可以在域内的任何节点上执行，因为更新该周期的步骤会在整个集群中传播更改。但是，radosgw-admin zone 需要在区域内的主机上执行。

5.11.2.1. 创建区组

创建 zone group 包括指定 zone group name。创建区域将假定它将位于默认域中，除非指定了 --rgw-realm=REALM_NAME。如果 zonegroup 是默认 zonegroup，请指定 --default 标志。如果 zonegroup 是 master zonegroup，请指定 --master 标志。

语法

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



注意

使用 zonegroup modify --rgw-zonegroup=ZONE_GROUP_NAME 修改现有 zone group 的设置。

5.11.2.2. 将区组设为默认值

zonegroups 列表中的一个 zonegroup 应当是末日 zonegroup。可能只有一个默认 zonegroup。如果只有一个 zonegroup，并且它没有在创建时指定为默认 zonegroup，则使其成为默认 zonegroup。另外，要更改哪个 zonegroup 是默认 zonegroup，请运行以下命令：

示例

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

注意

当 zonegroup 为默认值时，命令行假设 --rgw-zonegroup=ZONE_GROUP_NAME 作为参数。

然后，更新周期：

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

5.11.2.3. 在区组中添加一个区

要向 zonegroup 添加一个区域，您必须在区域中的主机上运行此命令。要在 zonegroup 中添加区，请运行以下命令：

语法

```
radosgw-admin zonegroup add --rgw-zonegroup=ZONE_GROUP_NAME --rgw-zone=ZONE_NAME
```

然后，更新周期：

示例

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

5.11.2.4. 从区组中删除区

要从 zonegroup 中删除区，请运行以下命令：

语法

```
radosgw-admin zonegroup remove --rgw-zonegroup=ZONE_GROUP_NAME --rgw-
zone=ZONE_NAME
```

然后，更新周期：

示例

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

5.11.2.5. 重命名一个区组

要重命名 zonegroup，请运行以下命令：

语法

```
radosgw-admin zonegroup rename --rgw-zonegroup=ZONE_GROUP_NAME --zonegroup-new-
name=NEW_ZONE_GROUP_NAME
```

然后，更新周期：

示例

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

5.11.2.6. 删除 Zone 组

要删除 zonegroup，请运行以下命令：

语法

```
radosgw-admin zonegroup delete --rgw-zonegroup=ZONE_GROUP_NAME
```

然后，更新周期：

示例

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

5.11.2.7. 列出区组

Ceph 集群包含 zone group 的列表。要列出区组，请运行以下命令：

```
[ceph: root@host01 /]# radosgw-admin zonegroup list
```

radosgw-admin 返回一个 JSON 格式的 zone group 列表。

```
{
    "default_info": "90b28698-e7c3-462c-a42d-4aa780d24eda",
    "zonegroups": [
        "us"
    ]
}
```

5.11.2.8. 获取区组

要查看 zone group 的配置，请运行以下命令：

语法

```
radosgw-admin zonegroup get [--rgw-zonegroup=ZONE_GROUP_NAME]
```

zone group 配置类似如下：

```
{
    "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. 设置区组

定义 zone group 包括创建 JSON 对象，至少指定所需的设置：

1. name ：zone group 的名称。必需。

2. api_name ：zone group 的 API 名称。可选。

3. is_master ：确定 zone group 是 master zone group。必需。

   注意：您只能有一个 master zone group。

4. endpoint ：zone group 中所有端点的列表。例如，您可以使用多个域名来引用同一 zone group。请注意，需要使用正斜杠(\/)进行转意。您还可以为每个端点指定一个端口(fqdn:port)。可选。

5. hostnames ：zone group 中所有主机名的列表。例如，您可以使用多个域名来引用同一 zone group。可选。rgw dns name 设置将自动包含在此列表中。您应在更改此设置后重新启动网关守护进程。

6. master_zone ： zone group 的 master zone。可选。如果未指定，则使用默认区域。

   注意

   每个 zone group 只能有一个 master zone。

7. zones：zone group 中所有 zone 的列表。每个区域都有一个名称（必需），一个端点列表

（可选），以及网关是否记录元数据和数据操作（默认为false）。

8. *placement_targets ：放置目标列表（可选）。每个放置目标都包含放置目标的名称（必需）和一个标签列表（可选），以便只有具有标签的用户才能使用放置目标（例如，用户 info 中的 placement_tags 字段）。*

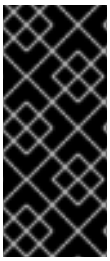
9. *default_placement ：对象索引和对象数据的默认放置目标。默认设置为 default-placement。您还可以在用户信息中为每个用户设置默认放置位置。*

要设置区组，创建一个由所需字段组成的 JSON 对象，将对象保存到文件中，如 zonegroup.json; 然后运行以下命令：

示例

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

其中 zonegroup.json 是您创建的 JSON 文件。



重要

默认 zone group is_master 默认设置为 true。如果您创建新 zone group 并希望使其成为 master zone group，您必须将 default zone group is_master 设置设置为 false，或删除 default zone group。

最后，更新周期：

示例

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

5.11.2.10. 设置区组映射

设置 zone group map 包括创建一个由一个或多个 zone group 组成的 JSON 对象，并为集群设置 master_zonegroup。zone group map 中的每个 zone group 由一个键/值对组成，其中 键 设置等同于单个 zone group 配置的 name 设置，而 val 则是由单个 zone group 配置组成的 JSON 对象。

您只能有一个 zone group is_master 等于 true，它必须指定为 zone group map 末尾的 master_zonegroup。以下 JSON 对象是默认 zone group map 的示例：

```
{
    "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
   }
}
```

要设置区组映射，请运行以下命令：

示例

```
[ceph: root@host01 /]# radosgw-admin zonegroup-map set --infile zonegroupmap.json
```

其中 zonegroupmap.json 是您创建的 JSON 文件。确保为 zone group map 中指定的区域创建了区域。最后，更新周期。

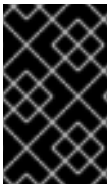
示例

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

5.11.3. Zones

Ceph 对象网关支持区域的概念。zone 定义由一个或多个 Ceph 对象网关实例组成的逻辑组。

配置区域与典型配置过程有所不同，因为并非所有设置都最终在 Ceph 配置文件中。您可以列出区域、获取区配置和设置区配置。



重要

所有 radosgw-admin zone 操作需要在 zone 中运行的主机上执行。

5.11.3.1. 创建区域

要创建区域，请指定区域名称。如果是一个 master 区域，指定 --master 选项。zone group 中只有一个 zone 可以是 master zone。要将区域添加到 zonegroup，请使用 zonegroup 名称指定 --rgw-zonegroup 选项。

重要

必须在位于 区域内的 Ceph 对象网关节点上创建区域。

语法

```
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
```

然后，更新周期：

示例

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

5.11.3.2. 删除区

要删除某个区域，请先将其从 zonegroup 中删除。

流程

1. **从 zonegroup 中删除区：**

   语法

   ```
   radosgw-admin zonegroup remove --rgw-zonegroup=ZONE_GROUP_NAME\
                                  --rgw-zone=ZONE_NAME
   ```

2. **更新周期：**

   示例

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

3. **删除区：**

   重要

   此过程 必须在区域内的主机上使用。

语法

```
radosgw-admin zone delete --rgw-zone=ZONE_NAME
```

4. *更新周期：*

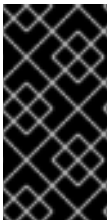
示例

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

重要

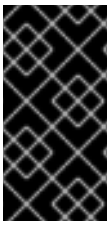
不要先从 zone group 中删除区域。否则，更新周期将失败。

如果已删除区域的池不会在其他任何位置使用，请考虑删除池。将以下示例中的 DELETED_ZONE_NAME 替换为已删除区的名称。



重要

当 Ceph 删除 zone 池后，它会以无法恢复的方式删除其中的所有数据。仅当 Ceph 客户端不再需要池内容时，仅删除 zone 池。

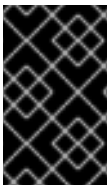


重要

在多域集群中，删除 .rgw.root 池以及 zone 池将移除群集的所有域信息。在删除 .rgw.root 池之前，确保. rgw.root 不包含其他活动域。

语法

```
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

删除池后，重新启动 RGW 流程。

5.11.3.3. 修改区域

若要修改区域，请指定区域名称和您要修改的参数。

重要

应在位于 区域内的 Ceph 对象网关节点上修改区域。

语法

```
radosgw-admin zone modify [options]

--access-key=<key>
--secret/--secret-key=<key>
--master
--default
--endpoints=<list>
```

然后，更新周期：

示例

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

5.11.3.4. 列出区域

以 root 用户身份列出集群中的区，请运行以下命令：

示例

```
[ceph: root@host01 /]# radosgw-admin zone list
```

5.11.3.5. 获取区域

以 root 用户身份获取区配置，请运行以下命令：

语法

```
radosgw-admin zone get [--rgw-zone=ZONE_NAME]
```

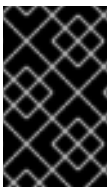
default 区类似如下：

```
{ "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. 设置区域

配置区域涉及指定一系列 Ceph 对象网关池。为保持一致性，我们建议使用与区域名称相同的池前缀。有关配置 池的详细信息，请参阅 Red Hat Ceph Storage 策略指南中的 池章节。



重要

应在位于 区域内的 Ceph 对象网关节点上设置区域。

要设置区，创建一个由池组成的 JSON 对象，将对象保存到文件中，如 zone.json; 然后运行以下命令，将 ZONE_NAME 替换为区的名称：

示例

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

其中 zone.json 是您创建的 JSON 文件。

然后，以 root 用户身份更新周期：

示例

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

5.11.3.7. 重命名区域

要重命名区域，请指定区域名称和新区域名称。在区中的主机上运行以下命令：

语法

```
radosgw-admin zone rename --rgw-zone=ZONE_NAME --zone-new-name=NEW_ZONE_NAME
```

然后，更新周期：

示例

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

第 6 章 高级配置

作为存储管理员，您可以配置 Ceph 对象网关的一些更高级功能。您可以配置多站点 Ceph 对象网关，并将其与目录服务（如 Microsoft Active Directory 和 OpenStack Keystone 服务）集成。

先决条件

- 一个正常运行的 Red Hat Ceph Storage 集群。

6.1. 配置 LDAP 和 CEPH 对象网关

执行以下步骤配置红帽目录服务器，以对 Ceph 对象网关用户进行身份验证。

6.1.1. 安装红帽目录服务器

Red Hat Directory Server 应该安装在带有图形用户界面(GUI)的 Red Hat Enterprise Linux 9 中，以便使用 Java Swing GUI Directory 和管理控制台。但是，红帽目录服务器仍可以从命令行界面(CLI)单独提供服务。

先决条件

- Red Hat Enterprise Linux(RHEL)安装在服务器上。
- Directory 服务器节点的 FQDN 可使用 DNS 或 /etc/hosts 文件解析。
- 将 Directory Server 节点注册到红帽订阅管理服务。
- 您的红帽帐户中提供了有效的红帽目录服务器订阅。

流程

- 按照红帽目录服务器安装指南的第 1 章和第 2 章中的内容进行操作。

其它资源

- 如需了解更多详细信息，请参阅 Red Hat Director 服务器安装指南 。

6.1.2. 配置目录服务器防火墙

在 LDAP 主机上，确保防火墙允许访问目录服务器的安全(636)端口，以便 LDAP 客户端可以访问 Directory 服务器。使默认非安全端口(389)保持关闭。

```
# firewall-cmd --zone=public --add-port=636/tcp
# firewall-cmd --zone=public --add-port=636/tcp --permanent
```

6.1.3. SELinux 的标签端口

为确保 SELinux 不阻止请求，请标记 SELinux 的端口。详情请查看红帽目录服务器 10 管理指南中的更改目录服务器端口号一节。

6.1.4. 配置 LDAPS

Ceph 对象网关使用简单的 ID 和密码与 LDAP 服务器进行身份验证，因此连接需要 LDAP 的 SSL 证书。要为 LDAP 配置目录服务器，请参阅 Red Hat Directory Server 11 管理指南中的 配置安全连接 章节。

LDAP 运行后，配置 Ceph 对象网关服务器，以信任目录服务器的证书。

1. 为签署 LDAP 服务器的 SSL 证书的证书颁发机构(CA)提取/下载 PEM 格式的证书。

2. 确认 /etc/openldap/ldap.conf 没有设置 TLS_REQCERT。

3. 确认 /etc/openldap/ldap.conf 包含 TLS_CACERTDIR /etc/openldap/certs 设置。

4. 使用 certutil 命令将 AD CA 添加到位于 /etc/openldap/certs 的存储中。例如，如果 CA 是 "msad-frog-MSAD-FROG-CA"，且 PEM 格式的 CA 文件为 ldap.pem，请使用以下命令：

   示例

```
# certutil -d /etc/openldap/certs -A -t "TC,," -n "msad-frog-MSAD-FROG-CA" -i
/path/to/ldap.pem
```

5. **在所有远程 LDAP 站点更新 SELinux：**

示例

```
# setsebool -P httpd_can_network_connect on
```

注意

即使 SELinux 处于 permissive 模式，这仍需要设置。

6. **使 certs 数据库全局可读：**

示例

```
# chmod 644 /etc/openldap/certs/*
```

7. **以非 root 用户身份使用"ldapwhoami"命令连接到服务器。**

示例

```
$ ldapwhoami -H ldaps://rh-directory-server.example.com -d 9
```

-d 9 选项将提供调试信息，以防 SSL 协商出现问题。

6.1.5. 检查网关用户是否存在

在创建网关用户之前，请确保 Ceph 对象网关还没有用户。

示例

```
[ceph: root@host01 /]# radosgw-admin metadata list user
```

用户名不应在此用户列表中。

6.1.6. 添加网关用户

创建 Ceph 对象网关用户到用户 LDAP。

流程

1. *为 Ceph 对象网关创建 LDAP 用户，并记下 binddn。由于 Ceph 对象网关使用 ceph 用户，因此请考虑使用 ceph 作为用户名。用户需要具有搜索目录的权限。Ceph 对象网关按照 rgw_ldap_binddn 中指定的绑定到此用户。*

2. *测试以确保用户创建有效。其中 ceph 是 People 下的用户 ID，example.com 是您可以在其中搜索用户的域。*

```
# 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. *在每个网关节点上，为用户的机密创建一个文件。例如，机密可能存储在具有 /etc/bindpass 的文件中。为安全起见，请将此文件的所有者更改为 ceph 用户和组，以确保它不能全局可读。*

4. **添加 rgw_ldap_secret 选项：**

语法

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

示例

```
[ceph: root@host01 /]# ceph config set client.rgw rgw_ldap_secret /etc/bindpass
```

5. **将绑定密码文件修补到 Ceph 对象网关容器，并重新应用 Ceph 对象网关规格：**

示例

```
service_type: rgw
service_id: rgw.1
service_name: rgw.rgw.1
placement:
  label: rgw
  extra_container_args:
  - -v
  - /etc/bindpass:/etc/bindpass
```

注意

Red Hat Ceph Storage 不会自动提供 /etc/bindpass，您需要确保内容在所有可能的 Ceph 对象网关实例节点上可用。

6.1.7. 将网关配置为使用 LDAP

1. 在所有 Ceph 节点上使用以下命令更改 Ceph 配置：

语法

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

示例

```
[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. 重新启动 Ceph 对象网关。

注意

使用 ceph orch ps 命令的输出，在 NAME 列下获取 SERVICE_TYPE。ID 信息。

a. 在存储集群中的单个节点上重启 Ceph 对象网关：

语法

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

示例

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

b. **在存储集群的所有节点上重启 Ceph 对象网关：**

语法

```
ceph orch restart SERVICE_TYPE
```

示例

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

6.1.8. 使用自定义搜索过滤器

您可以使用 rgw_ldap_searchfilter 设置创建自定义搜索过滤器来限制用户访问。使用 rgw_ldap_searchfilter 设置的方法有两种：

1. **指定部分过滤器：**

示例

```
"objectclass=inetorgperson"
```

Ceph 对象网关使用令牌的用户名和 rgw_ldap_dnattr 的值生成搜索过滤器。构建的过滤器随后与 rgw_ldap_searchfilter 值中的部分过滤器组合。例如，用户名和设置会生成最终搜索过滤器：

示例

```
"(&(uid=joe)(objectclass=inetorgperson))"
```

只有 LDAP 目录中找到了用户 joe 时，才会授予访问权限，具有 inetorgperson 的对象类并指定有效的密码。

2. 指定完整的过滤器：

完整的过滤器必须包含 USERNAME 令牌，该令牌在身份验证尝试期间替换为用户名。本例中不使用 rgw_ldap_dnattr 设置。例如，要将有效用户限制为特定组，请使用以下过滤器：

示例

```
"(&(uid=@USERNAME@)(memberOf=cn=ceph-users,ou=groups,dc=mycompany,dc=com))"
```

6.1.9. 将 S3 用户添加到 LDAP 服务器

在 LDAP 服务器上的管理控制台中，至少创建一个 S3 用户，以便 S3 客户端可以使用 LDAP 用户凭据。在将凭据传递给 S3 客户端时，记下要使用的用户名和机密。

6.1.10. 导出 LDAP 令牌

使用 LDAP 运行 Ceph 对象网关时，需要访问令牌。但是，访问令牌是从 access key 和 secret key 创建的。将 access key 和 secret key 导出为 LDAP 令牌。

1. 导出访问密钥：

语法

```
export RGW_ACCESS_KEY_ID="USERNAME"
```

2. 导出 secret 密钥：

语法

```
export RGW_SECRET_ACCESS_KEY="PASSWORD"
```

3. 导出令牌。对于 LDAP，使用 ldap 作为令牌类型(ttype)。

示例

```
radosgw-token --encode --ttype=ldap
```

对于 **Active Directory，使用 ad 作为令牌类型。**

示例

```
radosgw-token --encode --ttype=ad
```

结果是一个 **base-64** 编码字符串，即访问令牌。将此访问令牌提供给 **S3** 客户端，以代替 **access key。不再需要 secret** 密钥。

4. 可选：为方便起见，如果 **S3** 客户端使用环境变量，请将 **base-64** 编码字符串导出到 **RGW_ACCESS_KEY_ID** 环境变量。

示例

```
export
RGW_ACCESS_KEY_ID="ewogICAgIlJHV19UT0tFTiI6IHsKICAgICAgICAidmVyc2lvbiI6IDEsCi
AgICAgICAgInR5cGUiOiAibGRhcCIsCiAgICAgICAgImlkIjogImNlcGgiLAogICAgICAgICJrZXkiO
iAiODAwI0dvcmlsbGEiCiAgICB9Cn0K"
```

6.1.11. 使用 S3 客户端测试配置

使用 **Python Boto** 等脚本，使用 **Ceph** 对象网关客户端测试配置。

流程.

1. 使用 **RGW_ACCESS_KEY_ID** 环境变量配置 **Ceph** 对象网关客户端。或者，您可以复制 **base-64** 编码字符串，并将其指定为 **access key。以下是配置的 S3** 客户端的示例：

示例

```
cat .aws/credentials

[default]
aws_access_key_id =
ewogICaGbnjlwe9UT0tFTiI6IHsKICAgICAgICAidmVyc2lvbiI6IDEsCiAgICAgICAgInR5cGUiOiAi
YWQiLAogICAgICAgICJpZCI6ICJjZXBoIiwKICAgICAgICAia2V5IjogInBhc3M0Q2VwaCIKICAgI
H0KfQo=
aws_secret_access_key =
```

注意

不再需要 secret 密钥。

2. 运行 **aws s3 ls** 命令来验证用户：

示例

```
[root@host01 ~]# aws s3 ls --endpoint http://host03

2023-12-11 17:08:50 mybucket
2023-12-24 14:55:44 mybucket2
```

3. 可选： 您还可以运行 **radosgw-admin user** 命令来验证目录中的用户：

示例

```
[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. 配置 ACTIVE DIRECTORY 和 CEPH 对象网关

执行以下步骤，将 Active Directory 服务器配置为对 Ceph 对象网关用户进行身份验证。

6.2.1. 使用 Microsoft Active Directory

Ceph 对象网关 LDAP 身份验证与任何兼容 LDAP 的目录服务兼容，可针对简单绑定（包括 Microsoft Active Directory）进行配置。使用 Active Directory 与使用 RH Directory 服务器类似，Ceph 对象网关绑定为 rgw_ldap_binddn 设置中配置的用户，并使用 LDAP 来确保安全性。

配置 Active Directory 的过程基本上与配置 LDAP 和 Ceph 对象网关相同，但可能有一些特定于 Windows 的用法。

6.2.2. 为 LDAPS 配置 Active Directory

Active Directory LDAP 服务器默认配置为使用 LDAP。Windows Server 2012 及更高版本可以使用 Active Directory 证书服务。以下 MS TechNet 文章中提供了生成和安装与 Active Directory LDAP 一

起使用的 SSL 证书的说明：LDAP over SSL (LDAPS) Certificate。

注意

确保 Active Directory 主机上打开了 port 636。

6.2.3. 检查网关用户是否存在

在创建网关用户之前，请确保 Ceph 对象网关还没有用户。

示例

```
[ceph: root@host01 /]# radosgw-admin metadata list user
```

用户名不应在此用户列表中。

6.2.4. 添加网关用户

创建 Ceph 对象网关用户到用户 LDAP。

流程

1. 为 Ceph 对象网关创建 LDAP 用户，并记下 binddn。由于 Ceph 对象网关使用 ceph 用户，因此请考虑使用 ceph 作为用户名。用户需要具有搜索目录的权限。Ceph 对象网关按照 rgw_ldap_binddn 中指定的绑定到此用户。

2. 测试以确保用户创建有效。其中 ceph 是 People 下的用户 ID，example.com 是您可以在其中搜索用户的域。

```
# 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. 在每个网关节点上，为用户的机密创建一个文件。例如，机密可能存储在具有 /etc/bindpass

的文件中。为安全起见，请将此文件的所有者更改为 ceph 用户和组，以确保它不能全局可读。

4. **添加 rgw_ldap_secret 选项：**

语法

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

示例

```
[ceph: root@host01 /]# ceph config set client.rgw rgw_ldap_secret /etc/bindpass
```

5. **将绑定密码文件修补到 Ceph 对象网关容器，并重新应用 Ceph 对象网关规格：**

示例

```
service_type: rgw
service_id: rgw.1
service_name: rgw.rgw.1
placement:
  label: rgw
  extra_container_args:
  - -v
  - /etc/bindpass:/etc/bindpass
```

注意

Red Hat Ceph Storage 不会自动提供 /etc/bindpass，您需要确保内容在所有可能的 Ceph 对象网关实例节点上可用。

6.2.5. 将网关配置为使用 Active Directory

1. 设置 **rgw_ldap_secret** 设置后添加以下选项：

语法

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

示例

```
[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
```

对于 **rgw_ldap_uri** 设置，使用 **LDAP** 服务器的完全限定域名替换 **FQDN**。如果有多个 **LDAP** 服务器，请指定每个域。

对于 **rgw_ldap_binddn** 设置，将 **BINDDN** 替换为 **bind** 域。对于 **example.com** 域以及 用户和帐户 下的 **ceph** 用户，它应如下所示：

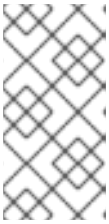
示例

```
rgw_ldap_binddn "uid=ceph,cn=users,cn=accounts,dc=example,dc=com"
```

*对于 **rgw_ldap_searchdn** 设置，请将 **SEARCHDN** 替换为搜索域。对于 **example.com** 域以及 **users** 和 **accounts** 下的用户，它应如下所示：*

```
rgw_ldap_searchdn "cn=users,cn=accounts,dc=example,dc=com"
```

2. *重启 **Ceph** 对象网关：*



注意

*使用 **ceph orch ps** 命令的输出，在 **NAME** 列下获取 **SERVICE_TYPE**。**ID** 信息。*

a. *在存储集群中的单个节点上重启 **Ceph** 对象网关：*

语法

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

示例

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

b. *在存储集群的所有节点上重启 **Ceph** 对象网关：*

语法

```
ceph orch restart SERVICE_TYPE
```

示例

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

6.2.6. 将 S3 用户添加到 LDAP 服务器

在 LDAP 服务器上的管理控制台中，至少创建一个 S3 用户，以便 S3 客户端可以使用 LDAP 用户凭据。在将凭据传递给 S3 客户端时，记下要使用的用户名和机密。

6.2.7. 导出 LDAP 令牌

使用 LDAP 运行 Ceph 对象网关时，需要访问令牌。但是，访问令牌是从 access key 和 secret key 创建的。将 access key 和 secret key 导出为 LDAP 令牌。

1. 导出访问密钥：

语法

```
export RGW_ACCESS_KEY_ID="USERNAME"
```

2. 导出 secret 密钥：

语法

```
export RGW_SECRET_ACCESS_KEY="PASSWORD"
```

3. 导出令牌。对于 **LDAP**，使用 **ldap** 作为令牌类型**(ttype)**。

示例

```
radosgw-token --encode --ttype=ldap
```

对于 **Active Directory**，使用 **ad** 作为令牌类型。

示例

```
radosgw-token --encode --ttype=ad
```

结果是一个 **base-64** 编码字符串，即访问令牌。将此访问令牌提供给 **S3** 客户端，以代替 **access key**。不再需要 **secret** 密钥。

4. 可选：为方便起见，如果 **S3** 客户端使用环境变量，请将 **base-64** 编码字符串导出到 **RGW_ACCESS_KEY_ID** 环境变量。

示例

```
export
RGW_ACCESS_KEY_ID="ewogICAgIlJHV19UT0tFTiI6IHsKICAgICAgICAidmVyc2lvbiI6IDEsCi
```

```
AgICAgICAgInR5cGUiOiAibGRhcCIsCiAgICAgICAgImlkIjogImNlcGgiLAogICAgICAgICJrZXkiOiAiODAwI0dvcmlsbGEiCiAgICB9Cn0K"
```

6.2.8. 使用 S3 客户端测试配置

使用 Python Boto 等脚本，使用 Ceph 对象网关客户端测试配置。

流程.

1. 使用 RGW_ACCESS_KEY_ID 环境变量配置 Ceph 对象网关客户端。或者，您可以复制 base-64 编码字符串，并将其指定为 access key。以下是配置的 S3 客户端的示例：

   示例

```
cat .aws/credentials

[default]
aws_access_key_id =
ewogICaGbnjlwe9UT0tFTiI6IHsKICAgICAgICAidmVyc2lvbiI6IDEsCiAgICAgICAgInR5cGUiOiAiYWQiLAogICAgICAgICJpZCI6ICJjZXBoIiwKICAgICAgICAia2V5IjogInBhc3M0Q2VwaCIKICAgIH0KfQo=
aws_secret_access_key =
```

   注意

   不再需要 secret 密钥。

2. 运行 aws s3 ls 命令来验证用户：

   示例

```
[root@host01 ~]# aws s3 ls --endpoint http://host03
```

```
2023-12-11 17:08:50 mybucket
2023-12-24 14:55:44 mybucket2
```

3. *可选： 您还可以运行 **radosgw-admin user** 命令来验证目录中的用户：*

示例

```
[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 对象网关和 OPENSTACK KEYSTONE

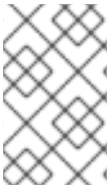
作为存储管理员，您可以使用 OpenStack 的 Keystone 身份验证服务通过 Ceph 对象网关对用户进行身份验证。在您可以配置 Ceph 对象网关之前，您需要先配置 Keystone。这将启用 Swift 服务，并将 Keystone 服务指向 Ceph 对象网关。接下来，您需要配置 Ceph 对象网关，以接受来自 Keystone 服务的身份验证请求。

先决条件

- 正在运行的 Red Hat OpenStack Platform 环境。
- 正在运行的 Red Hat Ceph Storage 环境。
- 正在运行的 Ceph 对象网关环境。

6.3.1. Keystone 身份验证的角色

OpenStack Keystone 服务提供三个角色： admin、member 和 reader。这些角色是分层的；具有 admin 角色的用户继承了 member 角色的功能，并且具有 member 角色的用户继承了 reader 角色的功能。



注意

member 角色的读取权限仅适用于它所属的项目的对象。

admin

admin 角色为特定范围内的最高授权级别保留。这通常包括资源或 API 上的所有创建、读取、更新或删除操作。

成员

默认情况下，不直接使用 member 角色。它在部署期间提供灵活性，并有助于减少管理员的责任。

例如，您可以使用默认 成员 角色和简单的策略覆盖覆盖部署的策略，以允许系统成员更新服务和端点。这提供了 admin 和 reader 角色之间的授权层。

读取器

reader 角色保留给只读操作，无论范围如何。

警告

如果您使用 读者 访问敏感信息，如镜像许可证密钥、管理镜像数据、管理卷元数据、应用程序凭证和 secret，您可能会意外公开敏感信息。因此，公开这些资源的 API 应该仔细考虑 reader 角色的影响，并适当地延迟对 member 和 admin 角色的访问。

6.3.2. Keystone 身份验证和 Ceph 对象网关

使用 OpenStack Keystone 验证用户身份的组织可以将 Keystone 与 Ceph 对象网关集成。Ceph 对象网关使网关能够接受 Keystone 令牌，对用户进行身份验证，然后创建对应的 Ceph 对象网关用户。当 Keystone 验证令牌时，网关将考虑用户经过身份验证。

优点

- 使用 Keystone 为用户分配 admin、member 和 reader 角色。
- 在 Ceph 对象网关中创建用户。
- 使用 Keystone 管理用户.
- Ceph 对象网关将定期查询 Keystone，以获取已撤销令牌的列表。

6.3.3. 创建 Swift 服务

在配置 Ceph 对象网关之前，请配置 Keystone，使 Swift 服务已启用并指向 Ceph 对象网关。

先决条件

- 一个正在运行的 Red Hat Ceph Storage 集群。
- 访问 Ceph 软件存储库。
- OpenStack 控制器节点的根级别访问权限.

流程

- 创建 Swift 服务：

```
[root@swift~]# openstack service create --name=swift --description="Swift Service" object-store
```

创建服务将回显服务设置。

表 6.1. 示例

字段	值
description	Swift 服务
enabled	true
id	37c4c0e79571404cb4644201a4a6e5ee
name	swift
type	object-store

6.3.4. 设置 Ceph 对象网关端点

在创建了 Swift 服务后，将服务指向 Ceph 对象网关。

先决条件

- 一个正在运行的 Red Hat Ceph Storage 集群。

- *访问 Ceph 软件存储库。*
- *在 Red Hat OpenStack Platform 17.0 环境中运行 Swift 服务。*

流程

- *创建指向 Ceph 对象网关的 OpenStack 端点：*

语法

```
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"
```

将 REGION_NAME 替换为网关的 zone group name 或 region 名称的名称。使用适合 Ceph 对象网关的 URL 替换 URL。

示例

```
[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"
```

字段	值
adminurl	**http://radosgw.example.com:8080/swift/v1**
id	**e4249d2b60e44743a67b5e5b38c18dd3**

字段	值
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

设置端点将输出服务端点设置。

6.3.5. 验证 Openstack 使用 Ceph 对象网关端点

创建 Swift 服务并设置端点后，请显示端点以确保所有设置都正确。

先决条件

- *一个正在运行的 Red Hat Ceph Storage 集群。*
- *访问 Ceph 软件存储库。*

流程

1. *列出 Swift 服务下的端点：*

```
[root@swift~]# openstack endpoint list --service=swift
```

2. *验证上一命令中列出的端点的设置：*

语法

```
[root@swift~]# openstack endpoint show ENDPOINT_ID
```

显示端点将回显端点设置和服务设置。

表 6.2. 示例

字段	值
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

其它资源

- 有关获取端点详情的更多信息，请参阅 Red Hat OpenStack 指南中的显示端点。

6.3.6. 配置 Ceph 对象网关以使用 Keystone SSL

转换 Keystone 使用的 OpenSSL 证书，配置 Ceph 对象网关以搭配 Keystone 使用。当 Ceph 对象网关与 OpenStack 的 Keystone 身份验证交互时，Keystone 将终止自签名 SSL 证书。

先决条件

- 一个正在运行的 Red Hat Ceph Storage 集群。
- 访问 Ceph 软件存储库。

流程

1. 将 OpenSSL 证书转换为 db 格式：

示例

```
[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. 在运行 Ceph 对象网关的节点中安装 Keystone 的 SSL 证书。或者，将可配置的 rgw_keystone_verify_ssl 设置的值设置为 false。

将 rgw_keystone_verify_ssl 设置为 false 表示网关不会尝试验证证书。

6.3.7. 配置 Ceph 对象网关以使用 Keystone 身份验证

配置 Red Hat Ceph Storage，以使用 OpenStack 的 Keystone 身份验证。

先决条件

- 一个正在运行的 Red Hat Ceph Storage 集群。
- 访问 Ceph 软件存储库。

- 具有生产环境的 admin 特权。

流程

1. 为每个网关实例执行以下操作：

   a. 将 thenss_db_path 设置设置为 NSS 数据库存储的路径：

   示例

```
[ceph: root@host01 /]# ceph config set client.rgw nss_db_path
"/var/lib/ceph/radosgw/ceph-rgw.rgw01/nss"
```

2. 提供身份验证凭证：

   可以为 OpenStack Identity API 为 keystone 配置 Keystone 服务租户、用户和密码，类似于系统管理员倾向于配置 OpenStack 服务的方式。提供用户名和密码可避免向 rgw_keystone_admin_token 设置提供共享 secret。

   重要

   红帽建议在生产环境中通过 admin 令牌禁用身份验证。服务租户凭据应当具有 admin 特权。

   所需的配置选项有：

   语法

```
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
```

示例

```
[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 对象网关用户映射到 Keystone 租户。Keystone 用户在 上分配有不同的角色，可能有多个租户。当 Ceph 对象网关获取票据时，它将查看租户，以及分配给该票据的用户角色，并且根据可配置的 rgw_keystone_accepted_roles 接受或拒绝请求。

其它资源

- 请参阅 Red Hat OpenStack Platform 的 用户和身份管理指南。

6.3.8. 重启 Ceph 对象网关守护进程

必须重新启动 Ceph 对象网关才能激活配置更改。

先决条件

- 一个正在运行的 Red Hat Ceph Storage 集群。
- 访问 Ceph 软件存储库。
- 生产环境的 admin特权。

流程

- 保存 Ceph 配置文件并将其分发到每个 Ceph 节点后，重启 Ceph 对象网关实例：

注意

使用 ceph orch ps 命令的输出，在 NAME 列下获取 SERVICE_TYPE。ID 信息。

a. 在存储集群中的单个节点上重启 Ceph 对象网关：

语法

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

示例

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

b. **在存储集群的所有节点上重启 Ceph 对象网关：**

语法

```
ceph orch restart SERVICE_TYPE
```

示例

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

第 7 章 安全性

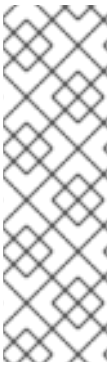
作为存储管理员，保护存储集群环境非常重要。Red Hat Ceph Storage 提供加密和密钥管理，以保护 Ceph 对象网关访问点的安全。

先决条件

- 一个正常运行的 Red Hat Ceph Storage 集群。
- 安装 Ceph 对象网关软件.

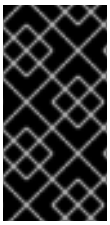
7.1. 服务器侧加密(SSE)

Ceph 对象网关支持为 S3 应用编程接口(API)上传对象的服务器端加密。服务器端加密意味着 S3 客户端以未加密的形式通过 HTTP 发送数据，而 Ceph 对象网关以加密的形式将数据存储在 Red Hat Ceph Storage 集群中。



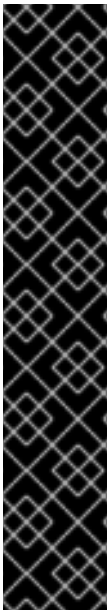
注意

- 红帽不支持静态大型对象(SLO)或动态大对象(DLO)的 S3 对象加密。
- 目前，没有 Server-Side 加密(SSE)模式支持 CopyObject。它目前正在开发 [BZ#2149450]。



重要

由于一个已知问题，服务器端加密与多站点复制不兼容。此问题将在以后的发行版本中解决。如需了解更多详细信息，请参阅 已知问题- Mult-site Object Gateway。



重要

若要使用加密，客户端请求需要通过 SSL 连接发送请求。除非 Ceph 对象网关使用 SSL，否则红帽不支持从客户端进行 S3 加密。但是，出于测试目的，管理员可以使用 ceph config set client.rgw 命令，在测试期间禁用 SSL，在运行时将 rgw_crypt_require_ssl 配置设置设置为 false，然后重新启动 Ceph 对象网关实例。

在生产环境中，可能无法通过 SSL 发送加密的请求。在这种情况下，使用 HTTP 和服务器端加密发送请求。

有关如何使用服务器端加密配置 HTTP 的详情，请参考下面的附加资源部分。

管理加密密钥有三个选项：

客户提供的键

在使用客户提供的密钥时，S3 客户端会传递加密密钥以及每个请求来读取或写入加密数据。客户负责管理这些密钥。客户必须记住用于加密每个对象的 Ceph 对象网关的关键是什么。

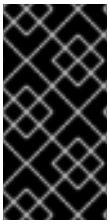
Ceph 对象网关根据 Amazon SSE-C 规范在 S3 API 中实施客户提供的关键行为。

由于客户处理密钥管理，并且 S3 客户端将密钥传递到 Ceph 对象网关，因此 Ceph 对象网关不需要特殊配置来支持这种加密模式。

密钥管理服务

在使用密钥管理服务时，安全密钥管理服务存储密钥，Ceph 对象网关则按需检索密钥，为数据加密或解密请求提供服务。

Ceph 对象网关根据 Amazon SSE-KMS 规范在 S3 API 中实施关键管理服务行为。



重要

目前，唯一测试的关键管理实施是 HashiCorp Vault 和 OpenStack Barbican。但是，OpenStack Barbican 是一个技术预览，不支持在生产环境中使用。

SSE-S3

使用 SSE-S3 时，密钥存储在密码库中，但它们由 Ceph 自动创建和删除，并在需要时检索以加密或解密数据。

Ceph 对象网关根据 Amazon SSE-S3 规范在 S3 API 中实施 SSE-S3 行为。

其它资源

- Amazon SSE-C
- Amazon SSE-KMS
- 配置服务器端加密
- HashiCorp Vault

7.1.1. 为现有 S3 存储桶设置默认加密

作为存储管理员，您可以为现有 Amazon S3 存储桶设置默认加密，以便在存储在存储桶时加密所有对象。您可以使用 Bucket 加密 API 支持 Amazon S3-managed 密钥(SSE-S3)或 Amazon KMS 客户主密钥(SSE-KMS)的服务器端加密。

注意

SSE-KMS 仅支持 Red Hat Ceph Storage 5.x，不支持 Red Hat Ceph Storage 4.x。

您可以使用 PutBucketEncryption API 管理现有 Amazon S3 存储桶的默认加密。在存储桶级别上定义默认加密，上传到此存储桶的所有文件都将具有此加密。

先决条件

- 一个正在运行的 Red Hat Ceph Storage 集群。

- 安装 Ceph 对象网关.
- 已创建 S3 存储桶。
- 创建的用户具有访问权限的 S3 用户。
- 使用安装的 AWS CLI 软件包访问 Ceph 对象网关客户端。

流程

1. 为加密配置创建 JSON 文件：

示例

```
[user@client ~]$ vi bucket-encryption.json
```

2. 在文件中添加加密配置规则：

示例

```
{
 "Rules": [
       {
    "ApplyServerSideEncryptionByDefault": {
                  "SSEAlgorithm": "AES256"
    }
      }
   ]
}
```

3. *为存储桶设置默认加密：*

语法

```
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_]
```

示例

```
[user@client ~]$ aws --endpoint-url=http://host01:80 s3api put-bucket-encryption --bucket testbucket --server-side-encryption-configuration file://bucket-encryption.json
```

验证

- *检索存储桶的存储桶加密配置：*

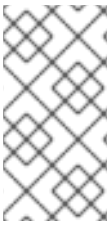
语法

```
aws --endpoint-url=pass:q[_RADOSGW_ENDPOINT_URL_]:pass:q[_PORT_] s3api get-bucket-encryption --bucket BUCKET_NAME
```

示例

```
[user@client ~]$  aws --profile ceph --endpoint=http://host01:80 s3api get-bucket-encryption --bucket testbucket
```

```
{
   "ServerSideEncryptionConfiguration": {
      "Rules": [
         {
            "ApplyServerSideEncryptionByDefault": {
               "SSEAlgorithm": "AES256"
            }
         }
      ]
   }
}
```



注意

如果存储桶没有默认的加密配置，get-bucket-encryption 命令会返回 ServerSideEncryptionConfigurationNotFoundError。

7.1.2. 删除默认存储桶加密

您可以使用 s3api delete-bucket-encryption 命令删除指定存储桶的默认存储桶加密。

先决条件

- 一个正在运行的 Red Hat Ceph Storage 集群。
- 安装 Ceph 对象网关.
- 已创建 S3 存储桶。
- 创建的用户具有访问权限的 S3 用户。
- 使用安装的 AWS CLI 软件包访问 Ceph 对象网关客户端。

流程

- ***删除存储桶加密配置：***

语法

```
aws --endpoint-url=RADOSGW_ENDPOINT_URL:PORT s3api delete-bucket-encryption --bucket BUCKET_NAME
```

示例

```
[user@client ~]$ aws --endpoint-url=http://host01:80 s3api delete-bucket-encryption --bucket testbucket
```

验证

- ***检索存储桶的存储桶加密配置：***

语法

```
aws --endpoint-url=RADOSGW_ENDPOINT_URL:PORT s3api get-bucket-encryption --bucket BUCKET_NAME
```

示例

```
[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. 服务器端加密请求

在生产环境中，客户端通常通过代理联系 Ceph 对象网关。此代理称为负载平衡器，因为它连接到多个 Ceph 对象网关。当客户端发送请求到 Ceph 对象网关时，负载平衡器会将这些请求路由到多个 Ceph 对象网关，从而分发工作负载。

在这种配置中，SSL 终止可能同时发生在负载平衡器和负载平衡器和多个 Ceph 对象网关之间。通信仅使用 HTTP。要设置 Ceph 对象网关以接受服务器端加密请求，请参阅配置服务器端加密。

7.3. 配置服务器端加密

在可能无法通过 SSL 发送加密请求的情况下，您可以设置服务器端加密加密来使用 HTTP 向 Ceph 对象网关发送请求。

此流程使用 HAProxy 作为代理和负载均衡器。

先决条件

- 对存储集群中所有节点的根级别访问权限。
- 一个正在运行的 Red Hat Ceph Storage 集群。
- 安装 Ceph 对象网关软件.
- 安装 HAProxy 软件.

流程

1. 编辑 haproxy.cfg 文件：

示例

```
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. *注释掉允许访问* **http 前端的行，并添加指令来指示 HAProxy 使用 https 前端：**

示例

```
#    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. 将 **rgw_trust_forwarded_https** 选项设置为 **true** ：

示例

```
[ceph: root@host01 /]# ceph config set client.rgw rgw_trust_forwarded_https true
```

4. 启用并启动 **HAProxy**：

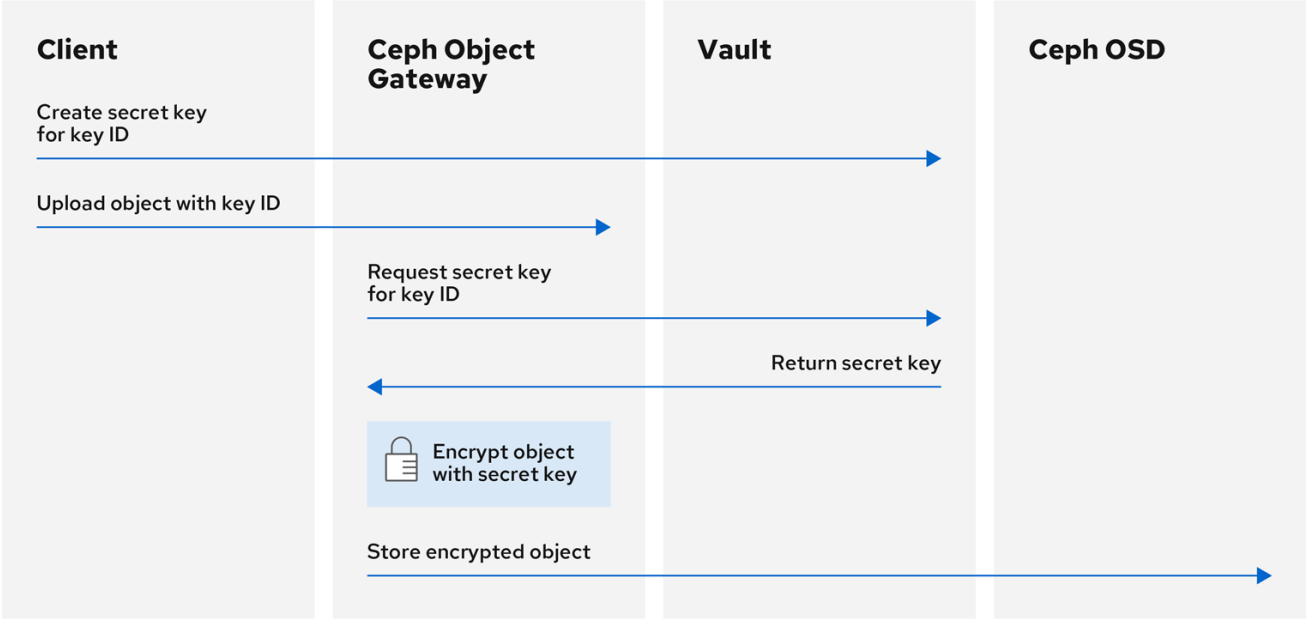
```
[root@host01 ~]# systemctl enable haproxy
[root@host01 ~]# systemctl start haproxy
```

其它资源

- 如需了解更多详细信息，请参阅 **Red Hat Ceph Storage** 对象网关指南中的 高可用性服务 部分。
- 如需了解更多详细信息，请参阅 **Red Hat Ceph Storage** 安装指南中的 **Red Hat Ceph Storage** 安装章节。

7.4. HASHICORP VAULT

作为存储管理员，您可以在 **HashiCorp Vault** 中安全地存储密钥、密码和证书，以用于 **Ceph** 对象网关。**HashiCorp Vault** 为 **Ceph** 对象网关使用的服务器端加密提供安全密钥管理服务。



基本工作流：

1. *客户端根据对象的密钥 ID 从 Vault 请求创建 secret key。*
2. *客户端将带有对象的密钥 ID 的对象上传到 Ceph 对象网关。*
3. *然后，Ceph 对象网关从 Vault 请求新创建的机密密钥。*
4. *Vault 通过将机密密钥返回到 Ceph 对象网关来回复请求。*
5. *现在，Ceph 对象网关可以使用新的机密密钥加密对象。*
6. *在加密完成后，对象存储在 Ceph OSD 上。*

重要

红帽与我们的技术合作伙伴合作，将本文档作为为客户提供服务。但是，红帽不提供对这个产品的支持。如果您需要此产品的技术协助，请联系 Hashicorp 以获得支持。

先决条件

- 一个正在运行的 Red Hat Ceph Storage 集群。
- 安装 Ceph 对象网关软件.
- 安装 HashiCorp Vault 软件.

7.4.1. Vault 的 secret 引擎

HashiCorp Vault 提供多个机密引擎来生成、存储或加密数据。应用编程接口(API)向机密引擎发送数据调用，要求对该数据采取行动，机密引擎返回该操作请求的结果。

Ceph 对象网关支持两个 HashiCorp Vault secret 引擎：

- Key/Value 版本 2
- Transit

Key/Value 版本 2

Key/Value secret 引擎将随机 secret 存储在 Vault 中的磁盘上。使用 kv 引擎的版本 2，键可以具有可配置的版本数。默认版本数量为 10。删除版本不会删除底层数据，而是将数据标记为已删除，从而允许取消删除的版本。您可以使用 API 端点或 destroy 命令永久删除版本的数据。要删除密钥的所有版本和元数据，您可以使用 metadata 命令或 API 端点。键名称必须是字符串，在使用命令行界面时引擎会将非字符串值转换为字符串。要保留非字符串值，请提供 JSON 文件或使用 HTTP 应用编程接口(API)。

注意

对于访问控制列表(ACL)策略，Key/Value secret 引擎可识别 创建和更新功能之间的区别。

Transit

Transit secret 引擎对传输中数据执行加密功能。Transit secret 引擎可以生成哈希值，可以是随机字节的来源，也可对数据进行签名和验证。在使用 Transit secret 引擎时，Vault 不会存储数据。Transit

secret 引擎允许将同一密钥用于多个目的，从而支持密钥生成。此外，传输机密引擎支持密钥版本控制。Transit secret 引擎支持这些关键类型：

aes128-gcm96

带有 128 位 AES 密钥和 96 位非ce 的 AES-GCM；支持加密、解密、密钥派生和聚合加密

aes256-gcm96

带有 256 位 AES 密钥和 96 位非ce 的 AES-GCM；支持加密、解密、密钥生成和聚合加密（默认）

chacha20-poly1305

ChaCha20-Poly1305，带有 256 位密钥；支持加密、解密、密钥加密和聚合加密

ed25519

Ed25519; 支持签名、签名验证和密钥生成

ecdsa-p256

使用 curve P-256 的 ECDSA；支持签名和签名验证

ecdsa-p384

使用 curve P-384 的 ECDSA；支持签名和签名验证

ecdsa-p521

使用 curve P-521 的 ECDSA；支持签名和签名验证

rsa-2048

2048 位 RSA 密钥；支持加密、解密、签名和签名验证

rsa-3072

3072 位 RSA 密钥；支持加密、解密、签名和签名验证

rsa-4096

4096 位 RSA 密钥；支持加密、解密、签名和签名验证

其它资源

- 如需更多信息，请参阅 Vault 项目网站上的 KV Secrets Engine 文档。

- 如需更多信息，请参阅 Vault 项目网站上的 Transit Secrets Engine 文档。

7.4.2. Vault 的身份验证

HashiCorp Vault 支持多种类型的身份验证机制。Ceph 对象网关目前支持 Vault 代理方法。Ceph 对象网关使用 rgw_crypt_vault_auth 和 rgw_crypt_vault_addr 选项来配置 HashiCorp Vault 的使用。

重要

红帽支持使用 Vault 代理作为容器的身份验证方法，容器上不支持使用令牌身份验证。

Vault 代理

Vault 代理是在客户端节点上运行的守护进程，提供客户端缓存以及令牌续订。Vault 代理通常在 Ceph 对象网关节点上运行。运行 Vault 代理并刷新令牌文件。在此模式中使用 Vault 代理时，您可以使用文件系统权限限制谁有权使用令牌。另外，Vault 代理也可以充当代理服务器，即 Vault 将在需要时添加令牌，并将其添加到传递给它的请求中，然后再将它们转发到实际服务器。Vault 代理仍然可以像在文件系统中存储令牌时一样处理令牌续订。需要保护 Ceph 对象网关用于连接 Vault 代理的网络，例如，Vault 代理仅侦听 localhost。

其它资源

- 如需更多信息，请参阅 Vault 项目站点上的 Vault 代理 文档。

7.4.3. Vault 的命名空间

将 HashiCorp Vault 用作企业服务，可为组织内的团队可以使用的隔离命名空间提供集中管理。这些隔离的命名空间环境称为 租户，组织内的团队可以利用这些租户将其策略、机密和身份与其他团队隔离。Vault 的命名空间功能帮助支持单一基础架构内的安全多租户。

其它资源

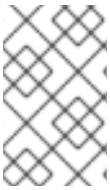
- 如需更多信息，请参阅 Vault 项目站点上的 Vault Enterprise 命名空间 文档。

7.4.4. 传递引擎兼容性支持

旧版 Ceph 的兼容性支持将 Transit 引擎用作简单的密钥存储。您可以使用 Transit 引擎中的 compat 选项来配置兼容性支持。您可以使用以下命令禁用之前的支持：

示例

```
[ceph: root@host03 /]# ceph config set client.rgw rgw_crypt_vault_secret_engine transit compat=0
```



注意

这是将来的版本的默认方法，您可以使用当前版本进行新的安装。

当前版本的正常默认设置为：

示例

```
[ceph: root@host03 /]# ceph config set client.rgw rgw_crypt_vault_secret_engine transit compat=1
```

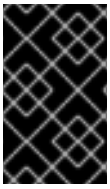
这可实现新创建的对象的新引擎，仍允许将旧引擎用于旧对象。若要访问旧对象和新对象，Vault 令牌必须具有旧的和新传输策略。

您可以使用以下命令强制使用旧引擎：

示例

```
[ceph: root@host03 /]# ceph config set client.rgw rgw_crypt_vault_secret_engine transit compat=2
```

如果 Vault 以 export/encryption-key 结束，则默认选择此模式。



重要

在配置了 client.rgw 选项后，您需要重启 Ceph 对象网关守护进程，使新值生效。

其它资源

- 如需更多信息，请参阅 Vault 项目站点上的 Vault 代理 文档。

7.4.5. 为 Vault 创建令牌策略

令牌策略指定所有 Vault 令牌的电源。一个令牌可以有多个策略。您应该在配置中使用所需的策略。

先决条件

- 一个正在运行的 Red Hat Ceph Storage 集群。
- 安装 HashiCorp Vault 软件。
- 对 HashiCorp Vault 节点的根级别访问权限。

流程

1. 创建令牌策略：

   a. 对于 Key/Value secret 引擎：

   示例

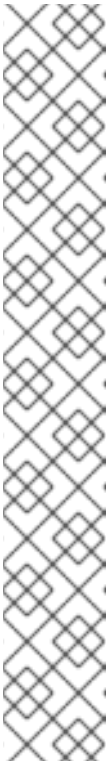
```
[root@vault ~]# vault policy write rgw-kv-policy -<<EOF
 path "secret/data/*" {
   capabilities = ["read"]
 }
EOF
```

b.

*对于 **Transit 引擎 :***

示例

```
[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
```



注意

如果您在旧版本的 Ceph 上使用了 Transit secret 引擎，则令牌策略为：

示例

```
[root@vault ~]# vault policy write old-rgw-transit-policy -<<EOF
  path "transit/export/encryption-key/*" {
    capabilities = ["read"]
  }
EOF
```

如果您使用 SSE-KMS 和 SSE-S3，您应该将每个指向单独的容器。您可以使用单独的 Vault 实例，或者在通用传输点下单独挂载实例或不同的分支。如果您不使用单独的 Vault 实例，您可以使用 rgw_crypt_vault_prefix 和 rgw_crypt_sse_s3_vault_prefix 将 SSE-KMS 或 SSE-S3 指向 serparate 容器。向 SSE-KMS bucket 所有者授予 Vault 权限时，您不应该为它们授予 SSE-S3 密钥的权限；只有 Ceph 应该有权访问 SSE-S3 密钥。

7.4.6. 将 Ceph 对象网关配置为使用 SSE-KMS 和 Vault

要将 Ceph 对象网关配置为使用带 SSE-KMS 的 HashiCorp Vault 进行密钥管理，它必须设置为加密密钥存储。目前，Ceph 对象网关支持两种不同的机密引擎，以及两种不同的身份验证方法。

先决条件

- 一个正在运行的 Red Hat Ceph Storage 集群。
- 安装 Ceph 对象网关软件.
- Ceph 对象网关节点的根级别访问权限.

流程

1. ***使用 ceph config set client.rgw OPTION VALUE 命令启用 Vault 作为加密密钥存储：***

语法

```
ceph config set client.rgw rgw_crypt_s3_kms_backend vault
```

2. ***添加以下选项和值：***

语法

```
ceph config set client.rgw rgw_crypt_vault_auth agent
ceph config set client.rgw rgw_crypt_vault_addr http://VAULT_SERVER:8100
```

3. ***根据用例自定义策略。***

4. ***获取 role-id：***

语法

```
vault read auth/approle/role/rgw-ap/role-id -format=json | \ jq -r .data.role_id >
PATH_TO_FILE
```

5. ***获取 secret-id：***

语法

```
vault read auth/approle/role/rgw-ap/role-id -format=json | \ jq -r .data.secret_id >
PATH_TO_FILE
```

6. *创建 **Vault** 代理的配置：*

示例

```
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. *使用 **systemctl** 运行持久性守护进程：*

示例

```
[root@host03 ~]# /usr/local/bin/vault agent -config=/usr/local/etc/vault/rgw-agent.hcl
```

8. *当 Vault 代理运行时，令牌文件填充有效令牌。*

9. *选择 Vault 机密引擎，可以是 Key/Value 或 Transit。*

   a. *如果使用 Key/Value，请添加以下行：*

   示例

   ```
   [ceph: root@host03 /]# ceph config set client.rgw rgw_crypt_vault_secret_engine kv
   ```

   b. *如果使用 Transit，请添加以下行：*

   示例

   ```
   [ceph: root@host03 /]# ceph config set client.rgw rgw_crypt_vault_secret_engine transit
   ```

10. *使用 ceph config set client.rgw OPTION VALUE 命令将 Vault 命名空间设置为检索加密密钥：*

    示例

    ```
    [ceph: root@host03 /]# ceph config set client.rgw rgw_crypt_vault_namespace
    testnamespace1
    ```

11. 通过设置路径前缀来限制 Ceph 对象网关从 Vault 中检索加密密钥的位置：

示例

```
[ceph: root@host03 /]# ceph config set client.rgw rgw_crypt_vault_prefix /v1/secret/data
```

a. 对于可导出的 Transit 键，请设置前缀路径，如下所示：

示例

```
[ceph: root@host03 /]# ceph config set client.rgw rgw_crypt_vault_prefix
/v1/transit/export/encryption-key
```

假设 Vault 服务器的域名是 vault-server，Ceph 对象网关将从以下 URL 获取加密传输密钥：

示例

```
http://vault-server:8200/v1/transit/export/encryption-key
```

12. 重新启动 Ceph 对象网关守护进程。

a.

在存储集群中的单个节点上重启 Ceph 对象网关：

语法

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

示例

```
[root@host03 ~]# systemctl restart ceph-c4b34c6f-8365-11ba-dc31-
529020a7702d@rgw.realm.zone.host01.gwasto.service
```

b.

在存储集群的所有节点上重启 Ceph 对象网关：

语法

```
ceph orch restart SERVICE_TYPE
```

示例

```
[ceph: root@host03 /]# ceph orch restart rgw
```

其它资源

-

如需了解更多详细信息，请参阅 Red Hat Ceph Storage Object Gateway Guide 中的 Vault Secret engines 部分。

- 如需了解更多详细信息，请参阅 Red Hat Ceph Storage 对象网关指南中的 Vault 身份验证部分。

7.4.7. 将 Ceph 对象网关配置为使用 SSE-S3 和 Vault

要将 Ceph 对象网关配置为使用带有 SSE-S3 的 HashiCorp Vault 进行密钥管理，它必须设置为加密密钥存储。目前，Ceph 对象网关仅使用 代理 身份验证方法。

先决条件

- 一个正在运行的 Red Hat Ceph Storage 集群。
- 安装 Ceph 对象网关软件.
- Ceph 对象网关节点的根级别访问权限.

流程

1. 登录到 Cephadm shell

   示例

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

2. 启用 Vault 作为 secret 引擎以检索 SSE-S3 加密密钥：

   语法

   -

```
ceph config set client.rgw rgw_crypt_sse_s3_backend vault
```

3. **要设置用于 SSE-S3 和 Vault 的验证方法，请配置以下设置：**

语法

```
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
```

示例

```
[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. **根据您的用例自定义策略，以设置 Vault 代理。**

b. **获取 role-id：**

语法

```
vault read auth/approle/role/rgw-ap/role-id -format=json | \ jq -r .rgw-ap-role-id >
PATH_TO_FILE
```

c.

*获取 **secret-id**：*

语法

```
vault read auth/approle/role/rgw-ap/role-id -format=json | \ jq -r .rgw-ap-secret-id >
PATH_TO_FILE
```

d.

*创建 **Vault** 代理的配置：*

示例

```
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.

*使用 **systemctl** 运行持久性守护进程：*

示例

```
[root@host01 ~]# /usr/local/bin/vault agent -config=/usr/local/etc/vault/rgw-agent.hcl
```

f. *当 Vault 代理运行时，令牌文件填充有效令牌。*

4. *设置 Vault 机密引擎，用于检索加密密钥，可以是 Key/Value 或 Transit。*

a. *如果使用 Key/Value，请设置以下内容：*

示例

```
[ceph: root@host01 /]# ceph config set client.rgw rgw_crypt_sse_s3_vault_secret_engine kv
```

b. *如果使用 Transit，请设置以下内容：*

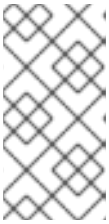
示例

```
[ceph: root@host01 /]# ceph config set client.rgw rgw_crypt_sse_s3_vault_secret_engine transit
```

5. *可选：配置 Ceph 对象网关以访问特定命名空间中的 Vault 以检索加密密钥：*

示例

```
[ceph: root@host01 /]# ceph config set client.rgw rgw_crypt_sse_s3_vault_namespace
company/testnamespace1
```



注意

Vault 命名空间允许团队在称为租户的隔离环境中操作。Vault 命名空间功能仅适用于 Vault Enterprise 版本。

6. **可选：通过设置 URL 路径前缀来限制对 Vault secret 空间的特定子集的访问，其中 Ceph 对象网关从中检索加密密钥：**

   a. **如果使用 Key/Value，请设置以下内容：**

   示例

   ```
   [ceph: root@host01 /]# ceph config set client.rgw rgw_crypt_sse_s3_vault_prefix
   /v1/secret/data
   ```

   b. **如果使用 Transit，请设置以下内容：**

   示例

   ```
   [ceph: root@host01 /]# ceph config set client.rgw rgw_crypt_sse_s3_vault_prefix
   /v1/transit
   ```

假设 Vault 服务器的域名是 vaultserver，Ceph 对象网关将从以下 URL 获取加密的传输密钥：

示例

```
http://vaultserver:8200/v1/transit
```

7. 可选：要使用自定义 SSL 认证通过 Vault 进行身份验证，请配置以下设置：

语法

```
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
```

示例

```
[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. 重新启动 Ceph 对象网关守护进程。

a. ***在存储集群中的单个节点上重启 Ceph 对象网关：***

语法

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

示例

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

b. ***在存储集群的所有节点上重启 Ceph 对象网关：***

语法

```
ceph orch restart SERVICE_TYPE
```

示例

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

其它资源

-

如需了解更多详细信息，请参阅 Red Hat Ceph Storage Object Gateway Guide 中的 Vault Secret engines 部分。

- 如需了解更多详细信息，请参阅 Red Hat Ceph Storage 对象网关指南中的 Vault 身份验证部分。

7.4.8. 使用 kv 引擎创建密钥

配置 HashiCorp Vault Key/Value secret 引擎(kv)，以便您可以创建用于 Ceph 对象网关的密钥。secret 作为键值对存储在 kv 机密引擎中。

重要

服务器端 encyption 的密钥长度必须为 256 位，并使用 base64 进行编码。

先决条件

- 一个正在运行的 Red Hat Ceph Storage 集群。
- 安装 HashiCorp Vault 软件。
- 对 HashiCorp Vault 节点的根级别访问权限。

流程

1. 启用 Key/Value 版本 2 secret 引擎：

示例

```
vault secrets enable -path secret kv-v2
```

2. 创建新密钥：

语法

```
vault kv put secret/PROJECT_NAME/BUCKET_NAME key=$(openssl rand -base64 32)
```

示例

```
[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. 使用传输引擎创建密钥

配置 HashiCorp Vault Transit 机密引擎（transit），以便您可以创建用于 Ceph 对象网关的密钥。使用 Transit secret 引擎创建密钥必须可以导出，才能使用 Ceph 对象网关进行服务器端加密。

先决条件

- 一个正在运行的 Red Hat Ceph Storage 集群。
- 安装 HashiCorp Vault 软件。
- 对 HashiCorp Vault 节点的根级别访问权限。

流程

1. 启用 **Transit secret** 引擎：

```
[root@vault ~]# vault secrets enable transit
```

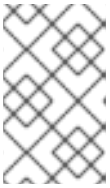
2. 创建新的可导出密钥：

语法

```
vault write -f transit/keys/BUCKET_NAME exportable=true
```

示例

```
[root@vault ~]# vault write -f transit/keys/mybucketkey exportable=true
```



注意

默认情况下，上述命令会创建一个 **aes256-gcm96** 类型密钥。

3. 验证密钥的创建：

语法

```
vault read transit/export/encryption-key/BUCKET_NAME/VERSION_NUMBER
```

示例

```
[root@vault ~]# vault read transit/export/encryption-key/mybucketkey/1

Key     Value
---     -----
keys    map[1:-gbTI9lNpqv/V/2lDcmH2Nq1xKn6FPDWarCmFM2aNsQ=]
name    mybucketkey
type    aes256-gcm96
```

注意

需要提供完整密钥路径，包括密钥版本。

7.4.10. 使用 AWS 和 Vault 上传对象

在上传对象到 Ceph 对象网关时，Ceph 对象网关将从 Vault 获取密钥，然后将对象加密并存储在 bucket 中。发出下载对象的请求时，Ceph 对象网关将自动从 Vault 检索对应的密钥并解密对象。若要上传对象，Ceph 对象网关从 Vault 获取密钥，然后加密对象并将其存储在 bucket 中。Ceph 对象网关从 Vault 检索对应的密钥，并在有下载对象的请求时解密对象。

注意

URL 使用基础地址（通过 rgw_crypt_vault_addr 选项和路径前缀设置）构建，该地址由 rgw_crypt_vault_prefix 选项设置。

先决条件

- 一个正在运行的 Red Hat Ceph Storage 集群。
- 安装 Ceph 对象网关软件.
- 安装 HashiCorp Vault 软件。

- 访问 **Ceph** 对象网关客户端节点.
- 访问 **Amazon Web Services(AWS)**。

流程

1. 使用 **AWS** 命令行客户端上传对象，并在请求中提供安全交换加密(SSE)密钥 **ID**：

   a. 对于 **Key/Value secret** 引擎：

   示例（使用 SSE-KMS）

   ```
   [user@client ~]$ aws --endpoint=http://radosgw:8000 s3 cp plaintext.txt
   s3://mybucket/encrypted.txt --sse=aws:kms --sse-kms-key-id myproject/mybucketkey
   ```

   示例（使用 SSE-S3）

   ```
   [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
   ```

   注意

   在示例中，**Ceph** 对象网关会从 **http://vault-server:8200/v1/secret/data/myproject/mybucketkey**获取 **secret**

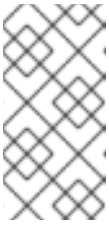
   b. 对于 **Transit** 引擎：

   示例（使用 SSE-KMS）

```
[user@client ~]$ aws --endpoint=http://radosgw:8000 s3 cp plaintext.txt
s3://mybucket/encrypted.txt --sse=aws:kms --sse-kms-key-id mybucketkey
```

示例（使用 SSE-S3）

```
[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
```



注意

在示例中，Ceph 对象网关会从 http://vaultserver:8200/v1/transit/mybucketkey 获取 secret

其它资源

- 如需更多信息，请参阅 Vault 项目站点上的 安装 Vault 文档。

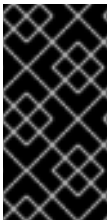
7.5. CEPH 对象网关和多因素身份验证

作为存储管理员，您可以管理 Ceph 对象网关用户的一次性密码(TOTP)令牌。

7.5.1. 多因素身份验证

当为对象版本控制配置 bucket 时，开发人员可以选择性地将存储桶配置为需要多因素身份验证(MFA)来删除请求。使用 MFA 时，基于时间的一次性密码(TOTP)令牌作为密钥传递给 x-amz-mfa 标头。令牌生成有虚拟 MFA 设备（如 Google Authenticator）或硬件 MFA 设备，如 Gemalto 提供的硬件 MFA 设备。

使用 adosgw-admin 为用户分配基于时间的一次性密码令牌。您必须设置一个 secret seed 和一个串行 ID。您还可以使用 adosgw-admin 列出、删除和重新同步令牌。



重要

在多站点环境中，建议将不同的令牌用于不同的区域，因为虽然 MFA ID 在用户的元数据上设置，但实际的 MFA 一次性密码配置驻留在本地区域的 OSD 上。

表 7.1. 术语

术语	描述
TOTP	基于时间的一次性密码.
Token serial	代表 TOTP 令牌 ID 的字符串。
Token seed	用于计算 TOTP 的 secret。它可以是十六进制或 base32。
TOTP seconds	用于 TOTP 生成的时间解析。
TOTP window	验证令牌时在当前令牌前后检查的 TOTP 令牌数量。
TOTP pin	TOTP 令牌在特定时间的有效值。

7.5.2. 创建用于多因素验证的 seed

要设置多因素身份验证(MFA)，您必须创建一个看到的 secret，供一次性密码生成器和后端 MFA 系统使用。

先决条件

- Linux 系统。
- 访问命令行 shell.

流程

1. 从 urandom Linux 设备文件中生成 30 个字符，并将其存储在 shell 变量 SEED 中：

示例

```
[user@host01 ~]$ SEED=$(head -10 /dev/urandom | sha512sum | cut -b 1-30)
```

2. 通过在 SEED 变量中运行 echo 来打印 seed:

示例

```
[user@host01 ~]$ echo $SEED
492dedb20cf51d1405ef6a1316017e
```

将一次性密码生成器和后端 MFA 系统配置为使用相同的 seed。

其它资源

- 如需更多信息，请参阅 Unable 为存储桶创建 RGW MFA 令牌。
- 有关更多信息，请参阅 Ceph 对象网关和多因素身份验证。

7.5.3. 创建新的多因素身份验证 TOTP 令牌

创建一个新的多因素身份验证(MFA)时间密码(TOTP)令牌。

先决条件

- 一个正在运行的 Red Hat Ceph Storage 集群。
- 已安装 Ceph 对象网关。
- 在 Ceph 监控节点上具有 root 访问权限。

- 生成了用于一次性密码生成器和 Ceph 对象网关 MFA 的 secret。

流程

- 创建新的 MFA TOTP 令牌：

语法

```
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
```

将 USERID 设置为设置 MFA 的用户名，将 SERIAL 设置为代表 TOTP 令牌 ID 的字符串，并将 SEED 设置为用于计算 TOTP 的十六进制或 base32 值。以下设置是可选的：将 SEED_TYPE 设置为 hex 或 base32，将 TOTP_SECONDS 设置为超时，或者将 TOTP_WINDOW 设置为验证令牌时要检查的 TOTP 令牌数量。

示例

```
[root@host01 ~]# radosgw-admin mfa create --uid=johndoe --totp-serial=MFAtest --totp-seed=492dedb20cf51d1405ef6a1316017e
```

其它资源

- 如需更多信息，请参阅为多因素身份验证创建 seed。
- 如需更多信息，请参阅 重新同步多因素身份验证令牌。

7.5.4. 测试多因素身份验证 TOTP 令牌

测试基于多因素身份验证(MFA)时间的一次性密码(TOTP)令牌。

先决条件

- 一个正在运行的 Red Hat Ceph Storage 集群。
- 已安装 Ceph 对象网关。
- 在 Ceph 监控节点上具有 root 访问权限。
- 使用 radosgw-admin mfa create 创建 MFA TOTP 令牌。

流程

- 测试 TOTP 令牌 PIN 以验证 TOTP 是否正常工作：

语法

```
radosgw-admin mfa check --uid=USERID --totp-serial=SERIAL --totp-pin=PIN
```

将 USERID 设置为设置 MFA 的用户名称，将 SERIAL 设置为代表 TOTP 令牌 ID 的字符串，然后将 PIN 设置为一次性密码生成器中的最新 PIN。

示例

```
[root@host01 ~]# radosgw-admin mfa check  --uid=johndoe --totp-serial=MFAtest --totp-pin=870305
ok
```

如果您是第一次测试 PIN，则可能会失败。如果失败，请重新同步令牌。请参阅 Red Hat Ceph Storage 对象网关配置和管理指南 中的 重新同步多因素身份验证令牌。

其它资源

- 如需更多信息，请参阅为多因素身份验证创建 seed。
- 如需更多信息，请参阅 重新同步多因素身份验证令牌。

7.5.5. 重新同步多因素身份验证 TOTP 令牌

重新同步多因素验证(MFA)基于时间的一次性密码令牌。

先决条件

- 一个正在运行的 Red Hat Ceph Storage 集群。
- 已安装 Ceph 对象网关。
- 在 Ceph 监控节点上具有 root 访问权限。
- 使用 radosgw-admin mfa create 创建 MFA TOTP 令牌。

流程

1. 在时间偏移或检查失败时，重新同步多因素验证 TOTP 令牌。

   这需要连续两个 pin：前一个 pin 和当前的 pin。

   语法

```
radosgw-admin mfa resync --uid=USERID --totp-serial=SERIAL --totp-pin=PREVIOUS_PIN --totp=pin=CURRENT_PIN
```

将 USERID 设置为设置 MFA 的用户名，将 SERIAL 设置为代表 TOTP 令牌 ID 的字符串，将 PREVIOUS_PIN 设置为用户的先前 PIN，并将 CURRENT_PIN 设置为用户的当前 PIN。

示例

```
[root@host01 ~]# radosgw-admin mfa resync --uid=johndoe --totp-serial=MFAtest --totp-pin=802021 --totp-pin=439996
```

2. *通过测试新 PIN 验证令牌是否已重新同步：*

语法

```
radosgw-admin mfa check --uid=USERID --totp-serial=SERIAL --totp-pin=PIN
```

将 USERID 设置为设置 MFA 的用户名称，将 SERIAL 设置为代表 TOTP 令牌 ID 的字符串，并将 PIN 设置为用户的 PIN。

示例

```
[root@host01 ~]# radosgw-admin mfa check  --uid=johndoe --totp-serial=MFAtest --totp-pin=870305
ok
```

其它资源

- 如需更多信息，请参阅 创建新的多因素身份验证 TOTP 令牌。

7.5.6. 列出多因素身份验证 TOTP 令牌

列出特定用户拥有的所有基于多因素身份验证(MFA)时间的一次性密码(TOTP)令牌。

先决条件

- 一个正在运行的 Red Hat Ceph Storage 集群。
- 已安装 Ceph 对象网关。
- 在 Ceph 监控节点上具有 root 访问权限。
- 使用 radosgw-admin mfa create 创建 MFA TOTP 令牌。

流程

- 列出 MFA TOTP 令牌：

  语法

  ```
  radosgw-admin mfa list --uid=USERID
  ```

  将 USERID 设置为设置 MFA 的用户名称。

  示例

  ```
  [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
        }
    ]
}
```

其它资源

- 如需更多信息，请参阅 创建新的多因素身份验证 TOTP 令牌。

7.5.7. 显示多因素身份验证 TOTP 令牌

通过指定串行，显示基于特定多因素验证(MFA)时间的一次性密码(TOTP)令牌。

先决条件

- 一个正在运行的 Red Hat Ceph Storage 集群。
- 已安装 Ceph 对象网关。
- 在 Ceph 监控节点上具有 root 访问权限。
- 使用 radosgw-admin mfa create 创建 MFA TOTP 令牌。

流程

- 显示 MFA TOTP 令牌：

语法

```
radosgw-admin mfa get --uid=USERID --totp-serial=SERIAL
```

将 USERID 设置为设置 MFA 的用户名，并将 SERIAL 设置为代表 TOTP 令牌 ID 的字符串。

其它资源

- 如需更多信息，请参阅 创建新的多因素身份验证 TOTP 令牌。

7.5.8. 删除多因素身份验证 TOTP 令牌

删除基于多因素身份验证(MFA)时间的一次性密码(TOTP)令牌。

先决条件

- 一个正在运行的 Red Hat Ceph Storage 集群。
- 已安装 Ceph 对象网关。
- 在 Ceph 监控节点上具有 root 访问权限。
- 使用 radosgw-admin mfa create 创建 MFA TOTP 令牌。

流程

1. 删除 MFA TOTP 令牌：

语法

```
radosgw-admin mfa remove --uid=USERID --totp-serial=SERIAL
```

*将 **USERID** 设置为设置 **MFA** 的用户名，并将 **SERIAL** 设置为代表 **TOTP** 令牌 **ID** 的字符串。*

示例

```
[root@host01 ~]# radosgw-admin mfa remove --uid=johndoe --totp-serial=MFAtest
```

2. *验证 **MFA TOTP** 令牌是否已删除：*

语法

```
radosgw-admin mfa get --uid=USERID --totp-serial=SERIAL
```

*将 **USERID** 设置为设置 **MFA** 的用户名，并将 **SERIAL** 设置为代表 **TOTP** 令牌 **ID** 的字符串。*

示例

```
[root@host01 ~]# radosgw-admin mfa get --uid=johndoe --totp-serial=MFAtest
MFA serial id not found
```

其它资源

- *有关更多信息，请参阅 Ceph 对象网关和多因素身份验证。*

第 8 章 管理

作为存储管理员，您可以使用 radosgw-admin 命令行界面(CLI)或使用 Red Hat Ceph Storage 控制面板管理 Ceph 对象网关。

注意

并非所有 Ceph 对象网关功能都可用于 Red Hat Ceph Storage 控制面板。

- 存储策略
- 无索引 Buckets
- 配置存储桶索引重新划分
- 压缩
- 用户管理
- 角色管理
- 配额管理
- bucket 管理
- 使用方法
- Ceph 对象网关数据布局

先决条件

- 一个正常运行的 Red Hat Ceph Storage 集群。
- 安装 Ceph 对象网关软件.

8.1. 创建存储策略

Ceph 对象网关通过标识放置目标并在与放置目标关联的池中存储 bucket 和对象来存储客户端存储桶和对象。如果您不配置放置目标，并将它们映射到实例区域配置中的池，Ceph 对象网关将使用默认目标和池，如 default_placement。

存储策略为 Ceph 对象网关客户端提供了一种访问存储策略的方式，即能够将特定类型的存储作为目标，如 SSD、SAS 驱动器和 SATA 驱动器，从而确保、持久性、复制和纠删代码。详情请参阅 Red Hat Ceph Storage 7 的存储策略指南。

要创建存储策略，请使用以下步骤：

1. 使用所需的存储策略，创建一个新的池 .rgw.buckets.special。例如，使用纠删代码、特定 CRUSH 规则集、副本数和 pg_num 和 pgp_num 计数自定义的池。
2. 获取 zone group 配置并将其存储在文件中：

语法

```
radosgw-admin zonegroup --rgw-zonegroup=ZONE_GROUP_NAME get > FILE_NAME.json
```

示例

```
[root@host01 ~]# radosgw-admin zonegroup --rgw-zonegroup=default get > zonegroup.json
```

3. *在 zonegroup.json 文件中，在 placement_target 下添加一个 special-placement 条目：*

示例

```
{
 "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. *使用修改的 zone group.json 文件设置 zone group ：*

示例

```
[root@host01 ~]# radosgw-admin zonegroup set < zonegroup.json
```

5. *获取区配置并将其存储在文件中，如 zone.json ：*

示例

示例

```
[root@host01 ~]# radosgw-admin zone get > zone.json
```

6. *编辑区文件并在 **placement_pool** 下添加新放置策略键：*

示例

```
{
 "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. 设置新区配置：

示例

```
[root@host01 ~]# radosgw-admin zone set < zone.json
```

8. 更新 zone group 映射：

示例

```
[root@host01 ~]# radosgw-admin period update --commit
```

special-placement 条目列为 placement_target。

9. 在发出请求时指定存储策略：

示例

```
$ 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"
```

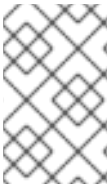
8.2. 创建无索引存储桶

您可以配置放置目标，其中创建的存储桶不使用存储桶索引来存储对象索引，即无索引存储桶。不使用数据复制或列表的放置目标可能会实现无索引存储桶。Indexless bucket 提供了一种机制，其中的放置目

标不会跟踪特定存储桶中的对象。这消除了每当发生对象写入时发生的资源争用，并减少 Ceph 对象网关对 Ceph 存储集群进行的往返次数。这可能会对并发操作和小对象写入性能产生积极的影响。

重要

bucket 索引不会反映存储桶的正确状态，列出这些存储桶将无法正确返回其对象列表。这会影响多个功能。具体来说，这些存储桶不会在多区环境中同步，因为存储桶索引不用于存储更改信息。红帽建议不要在无索引存储桶上使用 S3 对象版本控制，因为这个功能需要存储桶索引。



注意

使用无索引存储桶可移除单个 bucket 中最大对象数量的限值。

注意

无法从 NFS 列出无索引存储桶中的对象。

先决条件

- 正在运行的、健康的 Red Hat Ceph Storage 集群。
- 安装 Ceph 对象网关软件.
- Ceph 对象网关节点的根级别访问权限.

流程

1. 在 zonegroup 中添加新放置目标：

示例

```
[ceph: root@host03 /]# radosgw-admin zonegroup placement add --rgw-zonegroup="default"
\
  --placement-id="indexless-placement"
```

2. *在区中添加一个新的放置目标：*

示例

```
[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. *将 **zonegroup** 的默认放置设置为 **无索引放置** ：*

示例

```
[ceph: root@host03 /]# radosgw-admin zonegroup placement default --placement-id
"indexless-placement"
```

*在本例中，在 **indexless-placement** 目标中创建的存储桶将是无索引存储桶。*

4. *如果集群处于多站点配置中，请更新并提交周期：*

示例

```
[ceph: root@host03 /]# radosgw-admin period update --commit
```

5. 在存储集群的所有节点上重启 Ceph 对象网关，以使更改生效：

语法

```
ceph orch restart SERVICE_TYPE
```

示例

```
[ceph: root@host03 /]# ceph orch restart rgw
```

8.3. 配置存储桶索引重新划分

作为存储管理员，您可以在单一站点和多站点部署中配置存储桶索引重新划分，以提高性能。

您可以手动离线或动态在线对存储桶进行重新分片。

8.3.1. 存储桶索引重新分片

Ceph 对象网关将 bucket 索引数据存储在索引池中，默认为 .rgw.buckets.index 参数。当客户端将多个对象放在单个存储桶中，而不为每个存储桶的最大对象数量设置配额时，索引池可能会导致性能下降。

- 当您为每个存储桶添加大量对象时，存储桶索引重新划分可防止性能瓶颈。
- 您可以为新存储桶配置存储桶索引重新划分，或更改现有存储桶上的存储桶索引。

- 您需要将分片计数设置为计算的分片计数最接近的主数。作为主要数字的存储桶索引分片在分片之间的平均分布式存储桶索引条目中表现更好。

- bucket 索引可以手动重新划分或动态重新划分。

  在动态重新划分 bucket 索引的过程中，会定期检查所有 Ceph 对象网关存储桶，它会检测需要重新划分的存储桶。如果 bucket 增长大于 rgw_max_objs_per_shard 参数中指定的值，Ceph 对象网关会在后台动态重新定义存储桶。rgw_max_objs_per_shard 的默认值是每个分片 100k 对象。在升级的单站点配置上动态重新划分 bucket 索引，无需对 zone 或 zone group 进行任何修改。单个站点配置可以是以下任意一个：

  - 没有 realm 的默认区配置。
  - 至少一个 realm 的非默认配置。
  - 多域单站点配置。

8.3.2. 恢复存储桶索引

重新划分使用 bucket_index_max_shards = 0 创建的存储桶，移除存储桶的元数据。但是，您可以通过恢复受影响的存储桶来恢复存储桶。

/usr/bin/rgw-restore-bucket-index 工具在 /tmp 目录中创建临时文件。这些临时文件根据前面存储桶中的 bucket 对象计数来消耗空间。以前的带有超过 10M 对象的存储桶在 /tmp 目录中需要超过 4GB 的可用空间。如果 /tmp 中的存储空间耗尽，工具会失败并显示以下错误：

```
ln: failed to access '/tmp/rgwrbi-object-list.4053207': No such file or directory
```

临时对象被删除。

先决条件

- 一个正在运行的 Red Hat Ceph Storage 集群。

- *Ceph 对象网关至少安装两个站点。*
- *已安装 jq 软件包。*

流程

- *执行以下两个步骤之一来执行存储桶索引的恢复：*
  - *运行 radosgw-admin object reindex --bucket BUCKET_NAME --object OBJECT_NAME 命令。*
  - *运行脚本 - /usr/bin/rgw-restore-bucket-index -b BUCKET_NAME -p DATA_POOL_NAME。*

  示例

```
[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
```

注意

- 该工具不适用于版本控制存储桶。

```
[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.
```

- 工具的范围仅限于单一站点，而不是多站点，即如果我们在 site-1 上运行 rgw-restore-bucket-index 工具，它不会在 site-2 中恢复对象，反之亦然。在多站点上，应在存储桶的两个站点上执行恢复工具和对象 re-index 命令。

8.3.3. 存储桶索引重新划分的限制

重要

请谨慎使用以下限制：您的硬件选择会有影响，因此您应该始终与您的红帽客户团队讨论这些要求。

- 一个存储桶中的最大对象数量需要重新划分： 每个存储桶索引分片使用最多 102,400 个对象。要充分利用重新划分和最大化并行性，请在 Ceph 对象网关 bucket 索引池中提供足够数量的 OSD。这种并行化使用 Ceph 对象网关实例的数量进行扩展，并使用数字序列替换 in-order 索引分片枚举。默认锁定超时从 60 秒延长到 90 秒。

- 使用分片时的最大对象数量： 基于之前的测试，当前支持的存储桶索引分片数量为 65521。红帽质量保证未对存储桶分片执行完整的可扩展性测试。

- 使用分片时的最大对象数量： 基于之前的测试，当前支持的存储桶索引分片数量为 65,521。

- 您可以在其他区赶上前重新划分存储桶三次 ： 在旧的生成同步前，不建议重新划分存储桶。支持以前重新划分中的四个 bucket 生成。达到限制后，动态重新划分不会再次重新划分存储

桶，直到至少有一个旧日志生成被完全修剪为止。使用命令 **radosgw-admin bucket reshard** 会抛出以下错误：

> *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. 在简单部署中配置存储桶索引重新划分

要在所有新存储桶上启用和配置存储桶索引重新划分，请使用 **rgw_override_bucket_index_max_shards** 参数。

您可以将参数设置为以下值之一：

- **0** 禁用存储桶索引分片，这是默认值。
- 值大于 **0**，以启用存储桶分片并设置分片的最大数量。

先决条件

- 一个正在运行的 **Red Hat Ceph Storage** 集群。
- **Ceph** 对象网关至少安装两个站点。

流程

1. 计算推荐的分片数量：

> *number of objects expected in a bucket / 100,000*

注意

目前支持的存储桶索引分片的最大数量为 **65,521**。

2. 相应地设置 **rgw_override_bucket_index_max_shards** 选项：

语法

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

使用计算的推荐分片数量替换 **VALUE:**

示例

```
[ceph: root@host01 /]# ceph config set client.rgw rgw_override_bucket_index_max_shards
12
```

- 要为 **Ceph** 对象网关的所有实例配置 **bucket** 索引重新划分，请使用 **global** 选项设置 **rgw_override_bucket_index_max_shards** 参数。
- 若要仅为 **Ceph** 对象网关的特定实例配置 **bucket** 索引重新划分，请在实例下添加 **rgw_override_bucket_index_max_shards** 参数。

3. 在集群中的所有节点上重启 **Ceph** 对象网关以使更改生效：

语法

```
ceph orch restart SERVICE_TYPE
```

示例

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

其它资源

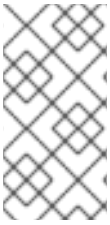
- *请参阅动态重新划分存储桶索引*
- *请参阅 手动重新划分存储桶索引*

8.3.5. 在多站点部署中配置存储桶索引重新划分

在多站点部署中，每个区域都可以有不同的 index_pool 设置来管理故障转移。要为一个 zone group 中的 zone 配置一致的分片计数，请在该 zone group 配置中设置 bucket_index_max_shards 参数。bucket_index_max_shards 参数的默认值为 11。

您可以将参数设置为以下值之一：

- *0 禁用存储桶索引分片。*
- *值大于 0，以启用存储桶分片并设置分片的最大数量。*



注意

将各个区域的索引池映射到基于 SSD 的 OSD 的 CRUSH 规则集也可能有助于 bucket 索引性能。如需更多信息，请参阅 建立性能域 部分。

重要

为防止多站点部署中同步问题，存储桶不应有超过三代差距。

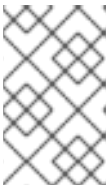
先决条件

- 一个正在运行的 **Red Hat Ceph Storage** 集群。
- **Ceph** 对象网关至少安装两个站点。

流程

1. 计算推荐的分片数量：

```
number of objects expected in a bucket / 100,000
```



注意

目前支持的存储桶索引分片的最大数量为 **65,521**。

2. 将 **zone group** 配置提取到 **zonegroup.json** 文件中：

示例

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

3. 在 **zonegroup.json** 文件中，为每个命名区设置 **bucket_index_max_shards** 参数：

语法

```
bucket_index_max_shards = VALUE
```

*使用计算的推荐分片数量替换 **VALUE:***

示例

```
bucket_index_max_shards = 12
```

4. *重置 **zone group:***

示例

```
[ceph: root@host01 /]# radosgw-admin zonegroup set < zonegroup.json
```

5. *更新周期：*

示例

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

6. *检查重新划分是否完成：*

语法

```
radosgw-admin reshard status --bucket BUCKET_NAME
```

示例

```
[ceph: root@host01 /]# radosgw-admin reshard status --bucket data
```

验证

- 检查存储集群的同步状态：

示例

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

8.3.6. 动态重新划分存储桶索引

您可以通过将存储桶添加到重新划分队列来动态重新划分存储桶索引。它被调度为重新划分。重新划分线程在后台运行，并一次执行调度的重新划分。

先决条件

- 一个正在运行的 Red Hat Ceph Storage 集群。
- Ceph 对象网关至少安装两个站点。

流程

1. 将 rgw_dynamic_resharding 参数设置为 true。

示例

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

2. **可选：使用以下命令自定义 Ceph 配置：**

语法

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

使用以下选项替换 OPTION：

- **rgw_reshard_num_logs ：重新划分日志的分片数量。默认值为 16。**
- **rgw_reshard_bucket_lock_duration ：在重新划分期间存储桶上锁定的持续时间。默认值为 360 秒。**
- **rgw_dynamic_resharding ：启用或禁用动态重新划分。默认值为 true。**
- **rgw_max_objs_per_shard ：每个分片的最大对象数量。默认值为每个分片 100000 对象。**
- **rgw_reshard_thread_interval ：重新线程处理循环之间的最长时间。默认值为 600 秒。**

示例

```
[ceph: root@host01 /]# ceph config set client.rgw rgw_reshard_num_logs 23
```

3. **将存储桶添加到重新划分队列中：**

语法

```
radosgw-admin reshard add --bucket BUCKET --num-shards NUMBER
```

示例

```
[ceph: root@host01 /]# radosgw-admin reshard add --bucket data --num-shards 10
```

4. **列出重新划分队列：**

示例

```
[ceph: root@host01 /]# radosgw-admin reshard list
```

5. **检查存储桶日志生成和分片：**

示例

```
[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. ***检查存储桶重新划分状态：***

语法

```
radosgw-admin reshard status --bucket BUCKET
```

示例

```
[ceph: root@host01 /]# radosgw-admin reshard status --bucket data
```

7. **立即处理重新划分队列中的条目：**

```
[ceph: root@host01 /]# radosgw-admin reshard process
```

8. **取消待处理的存储桶重新划分：**

警告

您只能取消 待处理的 重新划分操作。不要取消 正在进行的 重新划分的操作。

语法

```
radosgw-admin reshard cancel --bucket BUCKET
```

示例

```
[ceph: root@host01 /]# radosgw-admin reshard cancel --bucket data
```

验证

- *检查存储桶重新划分状态：*

  语法

  ```
  radosgw-admin reshard status --bucket BUCKET
  ```

  示例

  ```
  [ceph: root@host01 /]# radosgw-admin reshard status --bucket data
  ```

其他资源

- *请参阅 重新划分部分后清除存储桶条目的过时的实例，以移除过时的存储桶条目。*
- *请参阅 手动重新划分存储桶索引。*
- *请参阅在 简单部署中配置存储桶索引重新划分。*

8.3.7. 在多站点配置中动态重新划分存储桶索引

Red Hat Ceph Storage 支持在多站点配置中重新划分动态存储桶索引。该功能允许在多站点配置中重

新划分 bucket，而不中断其对象的复制。启用 rgw_dynamic_resharding 时，它会独立在每个区上运行，并且该区域可能会为同一 bucket 选择不同的分片计数。

这些步骤只需要遵循的步骤适用于现有的 Red Hat Ceph Storage 集群。在升级存储集群后，您需要在现有区和 zone group 上手动启用 重新划分 功能。

注意

默认情况下支持并启用 zone 和 zone group。

注意

您可以在其他区赶上前重新划分存储桶三次。如需了解更多详细信息，请参阅 存储桶索引重新划分的限制。

注意

如果创建存储桶并上传超过 阈值的对象，以便动态重新划分，您需要继续将 I/O 写入旧存储桶以开始重新划分过程。

先决条件

- 两个站点上的 Red Hat Ceph Storage 集群都会升级到最新的版本。
- 两个站点上启用的所有 Ceph 对象网关守护进程都升级到最新版本。
- 所有节点的根级别访问权限。

流程

1. 检查 zonegroup 上是否启用了 resharding ：

示例

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

-

*如果没有为 **zonegroup** 重新启用 **zonegroup** 功能来重新划分 **zonegroup**，然后继续操作。*

2. *在安装 **Ceph** 对象网关的多站点配置中，在所有 **zonegroups** 上启用 重新划分 功能：*

语法

```
radosgw-admin zonegroup modify --rgw-zonegroup=ZONEGROUP_NAME --enable-feature=resharding
```

示例

```
[ceph: root@host01 /]# radosgw-admin zonegroup modify --rgw-zonegroup=us --enable-feature=resharding
```

3. *更新周期和提交：*

示例

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

4. *在安装 **Ceph** 对象网关的多站点配置中，在所有区上启用 重新划分 功能：*

语法

```
radosgw-admin zone modify --rgw-zone=ZONE_NAME --enable-feature=resharding
```

示例

```
[ceph: root@host01 /]# radosgw-admin zone modify --rgw-zone=us-east --enable-feature=resharding
```

5. **更新周期和提交：**

示例

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

6. 验证 **zone 和 zonegroups 上是否启用了 重新划分 功能。您可以看到每个区都列出其 supported_features，zonegroups 会列出其 enabled_features**

示例

```
[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. *检查**同步状态**：*

示例

```
[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
```

在本例中，为 **us zonegroup** 启用了 **重新划分** 功能。

8. *可选：**您可以为 **zonegroups** 禁用 **重新划分** 功能：*

重要

要在任何 **singular zone** 上禁用重新划分，请将该特定区域的 **rgw_dynamic_resharding** 配置选项设置为 **false**。

a. 在安装了 **Ceph** 对象网关的多站点中所有 **zonegroups** 上禁用该功能：

语法

```
radosgw-admin zonegroup modify --rgw-zonegroup=ZONEGROUP_NAME --disable-feature=resharding
```

示例

```
[ceph: root@host01 /]# radosgw-admin zonegroup modify --rgw-zonegroup=us --disable-feature=resharding
```

b. 更新周期和提交：

示例

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

其它资源

- 有关动态 **bucket** 索引的更可配置的参数，请参阅 **Red Hat Ceph Storage Object Gateway Configuration and Administration Guide** 中的 **Resharding bucket** 索引动态 部分。

8.3.8. 手动重新划分存储桶索引

如果存储桶增长大于最佳初始配置，请使用 radosgw-admin bucket reshard 命令重新定义存储桶索引池。这个命令执行以下任务：

- 为指定存储桶创建新的 bucket 索引对象集合。
- 在这些 bucket 索引对象之间分发对象条目。
- 创建新的 bucket 实例。
- 使用存储桶链接新 bucket 实例，以便所有新的索引操作都通过新 bucket 索引进行。
- 将旧的和新的 bucket ID 打印到命令输出。

先决条件

- 一个正在运行的 Red Hat Ceph Storage 集群。
- Ceph 对象网关至少安装两个站点。

流程

1. 备份原始存储桶索引：

语法

```
radosgw-admin bi list --bucket=BUCKET > BUCKET.list.backup
```

示例

```
[ceph: root@host01 /]# radosgw-admin bi list --bucket=data > data.list.backup
```

2. **重新定义存储桶索引：**

语法

```
radosgw-admin bucket reshard --bucket=BUCKET --num-shards=NUMBER
```

示例

```
[ceph: root@host01 /]# radosgw-admin bucket reshard --bucket=data --num-shards=100
```

验证

- **检查存储桶重新划分状态：**

语法

```
radosgw-admin reshard status --bucket bucket
```

示例

```
[ceph: root@host01 /]# radosgw-admin reshard status --bucket data
```

其它资源

- 如需了解更多详细信息，请参阅 Red Hat Ceph Storage 对象网关指南中的 在多站点部署中配置存储桶索引重新划分。
- 请参阅动态重新划分存储桶索引。
- 请参阅在 简单部署中配置存储桶索引重新划分。

8.3.9. 重新划分后清理存储桶条目的过时的实例

重新划分过程可能无法自动清理存储桶条目的过时的实例，这些实例可能会影响存储集群的性能。

手动清理它们，以防止过时的实例对存储集群的性能造成负面影响。

重要

在清理过时的实例前，请联系红帽支持。

重要

仅在简单部署中使用这个步骤，而不在多站点集群中使用。

先决条件

- 一个正在运行的 Red Hat Ceph Storage 集群。

- 安装了 Ceph 对象网关.

流程

1. 列出过时的实例：

```
[ceph: root@host01 /]# radosgw-admin reshard stale-instances list
```

2. 清理存储桶条目的过时的实例：

```
[ceph: root@host01 /]# radosgw-admin reshard stale-instances rm
```

验证

- 检查存储桶重新划分状态：

语法

```
radosgw-admin reshard status --bucket BUCKET
```

示例

```
[ceph: root@host01 /]# radosgw-admin reshard status --bucket data
```

8.3.10. 启用压缩

Ceph 对象网关支持利用任何 Ceph 的压缩插件对上传对象进行服务器端压缩。它们是：

- **zlib ：支持。**
- **snappy ：支持。**
- **zstd ：支持。**

配置

要在区域的放置目标上启用压缩，请在 **radosgw-admin zone placement modify** 命令中提供 **--compression=TYPE** 选项。**compression TYPE** 指的是在编写新对象数据时要使用的压缩插件名称。

每个压缩对象存储压缩类型。更改设置不会影响解压缩现有压缩对象的能力，也不会强制 **Ceph** 对象网关重新压缩现有的对象。

此压缩设置适用于使用此放置目标上传到存储桶的所有新对象。

要禁用对区放置目标的压缩，请为 **radosgw-admin zone placement modify** 命令提供 **--compression=TYPE** 选项，并指定空字符串或 **none**。

示例

```
[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"
            }
        }
    ],
...
}
```

在启用或禁用压缩后，重新启动 Ceph 对象网关实例，以便更改生效。

注意

Ceph 对象网关创建 default 区域和一组池。对于生产环境部署，请参阅创建 Realm 部分。

Statistics

虽然所有现有命令和 API 都继续根据其未压缩数据报告对象和 bucket 大小，但 radosgw-admin bucket stats 命令包含所有存储桶的压缩统计信息。

radosgw-admin bucket stats 命令的使用类型是：

- rgw.main 指的是常规条目或对象。
- rgw.multimeta 是指未完成的多部分上传的元数据。
- rgw.cloud layer 指的是生命周期策略已过渡到云层的对象。当使用 retain_head_object=true 配置时，head 对象会保留不再包含数据，但仍然可以通过 HeadObject 请求提供对象的元数据。这些 stub 头对象使用 rgw.cloudlayer 类别。如需更多信息，请参阅 Red Hat Ceph Storage 对象网关指南中的 将数据转换到 Amazon S3 云服务 部分。

语法

```
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
        }
    },
...
}
```

其大小是存储桶中对象的积累大小，未压缩和未加密。size_kb 是以 KB 为单位的总大小，它计算为 ceiling (size/1024)。在本例中，它是 ceiling (1075028/1024)= 1050。

size_actual 是每个对象在一组 4096 字节块中分发后所有对象的累计大小。如果 bucket 有两个对象，大小为 4100 字节，另一个大小为 8500 字节，则第一个对象将向上舍入为 8192 字节，第二个对象舍入为 12288 字节，其存储桶的总数为 20480 字节。size_kb_actual 是实际大小（以 KB 为单位），计算为 size_actual/1024。在本例中，它是 1331200/1024 = 1300。

size_utilized 是压缩和/或加密后数据的总大小（以字节为单位）。加密可能会增加对象的大小，而压缩可能会减少它。size_kb_utilized 是以 KB 为单位的总大小，它计算为 ceiling (size_utilized/1024)。在本例中，它是 ceiling (592035/1024)= 579。

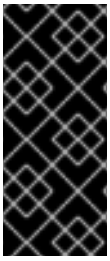
8.4. 用户管理

Ceph 对象存储用户管理是指属于 Ceph 对象存储服务的客户端应用的用户，而不是 Ceph 对象网关作为 Ceph 存储群集的客户端应用。您必须创建一个用户、访问密钥和机密，使客户端应用能够与 Ceph 对象网关服务交互。

用户类型有两种：

- 用户：术语"用户"反映了 S3 接口的用户。
- 子用户：术语"子用户"反映了 Swift 界面的用户。子用户会与用户关联.

您可以创建、修改、查看、暂停和删除用户和子用户。



重要

在多站点部署中管理用户时，ALWAYS 在 master zone group 的 master zone 中的 Ceph 对象网关节点上发出 radosgw-admin 命令，以确保用户在整个多站点集群中同步。不要从 second zone 或 second zone group 创建、修改或删除多站点集群上的用户。

除了创建用户和子用户 ID 外，您还可以为用户添加显示名称和电子邮件地址。您可以指定一个密钥和 secret，或者自动生成密钥和 secret。在生成或指定密钥时，请注意用户 ID 与 S3 密钥类型对应，子用户 ID 对应于 swift 密钥类型。Swift key 也具有 read, write, readwrite 和 full 的访问权限。

用户管理命令行语法通常遵循 用户 COMMAND USER_ID，其中 USER_ID 是 --uid= 选项，后跟用户的 ID (S3)或 --subuser= 选项，后跟用户名(Swift)。

语法

```
radosgw-admin user <create|modify|info|rm|suspend|enable|check|stats> <--uid=USER_ID|--subuser=SUB_USER_NAME> [other-options]
```

可能需要其他选项，具体取决于您发出的命令。

8.4.1. 多租户

Ceph 对象网关支持 S3 和 Swift API 的多租户，其中每个用户和 bucket 都位于"租户"。当多个租户使用通用存储桶名称，如 "test"、"main" 等时，多租户可防止命名空间冲突。

每个用户和 bucket 位于租户下。为向后兼容，将添加带有空名称的"传统"租户。每当不指定租户的情况下引用存储桶时，Swift API 将假定为"传统"租户。现有用户也存储在传统租户下，因此他们将像之前的版本一样访问 bucket 和对象。

租户对其没有任何操作。在管理用户时，它们根据需要显示和消失。为了创建、修改和删除具有显式租户的用户，提供了一个额外的选项 --tenant，或语法 "TENANT$USER" 用于 radosgw-admin 命令的参数。

要为 **S3** 创建用户 **testx$tester**，请运行以下命令：

示例

```
[root@host01 ~]# radosgw-admin --tenant testx --uid tester \
                --display-name "Test User" --access_key TESTER \
                --secret test123 user create
```

要为 **Swift** 创建用户 **testx$tester**，请运行以下命令之一：

示例

```
[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
```

注意

shell 中必须用引号括起具有显式租户的子用户。

8.4.2. 创建用户

使用 **user create** 命令创建 **S3-interface** 用户。您必须指定用户 **ID** 和显示名称。您也可以指定一个电子邮件地址。如果您不指定密钥或 **secret**，**radosgw-admin** 将自动为您生成。但是，如果您不希望使用生成的密钥/**secret** 对，您可以指定一个密钥和/或 **secret**。

语法

-

```
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
```

示例

```
[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

检查密钥输出。有时候，**radosgw-admin** 会生成 **JSON** 转义(\)字符，一些客户端不知道如何处理 **JSON** 转义字符。补救包括删除 **JSON** 转义字符(\)，用引号括起字符串，重新生成密钥以确保它没有 **JSON** 转义字符，或者手动指定密钥和 **secret**。

8.4.3. 创建子用户

要创建子用户（Swift 接口），您必须指定用户 ID(--uid=USERNAME)、子用户 ID 和子用户的访问级别。如果没有指定密钥或 secret，radosgw-admin 会自动为您生成它们。但是，如果您不希望使用生成的密钥和 secret 对，您可以指定一个密钥和 secret。

注意

full 并不是 readwrite，因为它还包含访问控制策略。

语法

```
radosgw-admin subuser create --uid=USER_ID --subuser=SUB_USER_ID --access=[ read | write | readwrite | full ]
```

示例

```
[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. 获取用户信息

要获取有关用户的信息，请指定 **user info** 和用户 **ID (--uid=USERNAME)**。

示例

```
[root@host01 ~]# radosgw-admin user info --uid=janedoe
```

若要获取租户用户的信息，可指定用户 **ID** 和租户名称。

```
[root@host01 ~]# radosgw-admin user info --uid=janedoe --tenant=test
```

8.4.5. 修改用户信息

要修改用户的信息，您必须指定用户 **ID (--uid=USERNAME)**和您要修改的属性。典型的修改包括密钥和 **secret**、电子邮件地址、显示名称和访问级别。

示例

```
[root@host01 ~]# radosgw-admin user modify --uid=janedoe --display-name="Jane E. Doe"
```

若要修改子用户值，指定 **subuser modify** 和子用户 **ID**。

示例

```
[root@host01 ~]# radosgw-admin subuser modify --subuser=janedoe:swift --access=full
```

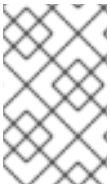
8.4.6. 启用和挂起用户

当您创建用户时，该用户默认是启用的。但是，您可以暂停用户特权并稍后重新启用它们。若要暂停用户，可指定 **user suspend** 和用户 **ID**。

```
[root@host01 ~]# radosgw-admin user suspend --uid=johndoe
```

要重新启用暂停的用户，请指定 **user enable** 和用户 **ID**：

```
[root@host01 ~]# radosgw-admin user enable --uid=johndoe
```



注意

禁用用户会禁用其子用户。

8.4.7. 删除用户

删除用户时，用户和子用户将从系统中删除。但是，如果需要，您只能删除子用户。要删除用户（及子用户），指定 **user rm** 和用户 **ID**。

语法

```
radosgw-admin user rm --uid=USER_ID[--purge-keys] [--purge-data]
```

示例

```
[ceph: root@host01 /]# radosgw-admin user rm --uid=johndoe --purge-data
```

若要仅删除子用户，指定 **subuser rm** 和 **subuser** 名称。

示例

```
[ceph: root@host01 /]# radosgw-admin subuser rm --subuser=johndoe:swift --purge-keys
```

选项包括：

- 清除数据： **--purge-data** 选项清除与 **UID** 关联的所有数据。
- 清除密钥： **--purge-keys** 选项清除与 **UID** 关联的所有密钥。

8.4.8. 删除子用户

移除子用户时，您将删除对 **Swift** 接口的访问。用户保留在系统中。要删除子用户，请指定 **subuser rm** 和 **subuser ID**。

语法

```
radosgw-admin subuser rm --subuser=SUB_USER_ID
```

示例

-

```
[root@host01 /]# radosgw-admin subuser rm --subuser=johndoe:swift
```

选项包括：

- 清除密钥： **--purge-keys** 选项清除与 UID 关联的所有密钥。

8.4.9. 重命名用户

要更改用户名称，使用 **radosgw-admin user rename** 命令。此命令花费的时间取决于用户拥有的 **bucket** 和对象的数量。如果数字较大，红帽建议使用 **screen** 软件包提供的 **Screen** 工具中的命令。

先决条件

- 正常运行的 **Ceph** 集群。
- 对运行 **Ceph** 对象网关的主机 的根 或 **sudo** 访问权限。
- 安装的 **Ceph** 对象网关。

流程

1. 重命名用户：

语法

```
radosgw-admin user rename --uid=CURRENT_USER_NAME --new-
uid=NEW_USER_NAME
```

示例

```
[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"
}
```

如果用户位于租户中，同时指定用户名和租户：

语法

```
radosgw-admin user rename --uid USER_NAME --new-uid NEW_USER_NAME --tenant
TENANT
```

示例

```
[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. *验证用户是否已成功重命名：*

语法

```
radosgw-admin user info --uid=NEW_USER_NAME
```

示例

```
[ceph: root@host01 /]# radosgw-admin user info --uid=user2
```

如果用户位于租户中，请使用 TENANT$USER_NAME 格式：

语法

```
radosgw-admin user info --uid= TENANT$USER_NAME
```

示例

```
[ceph: root@host01 /]# radosgw-admin user info --uid=test$user2
```

其它资源

- **screen(1) 手册页**

8.4.10. 创建密钥

要为用户创建密钥，您必须指定 **key create**。对于用户，指定用户 **ID** 和 **s3** 密钥类型。要为子用户创建密钥，您必须指定子用户 **ID** 和 **swift keytype**。

示例

```
[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. 添加和删除访问密钥

用户和子用户必须具有使用 S3 和 Swift 接口的访问密钥。当您创建用户或子用户且您没有指定 access key 和 secret 时，密钥和 secret 会自动生成。您可以创建一个密钥，并指定或生成 access key 和/或 secret。您也可以删除访问密钥和机密。选项包括：

- --secret=SECRET_KEY 指定 secret 密钥，例如手动生成的 secret 密钥。
- --gen-access-key 生成一个随机访问密钥（默认为 S3 用户）。
- --gen-secret 生成随机 secret key。
- --key-type=KEY_TYPE 指定密钥类型。这些选项有：swift 和 s3。

要添加密钥，请指定用户：

示例

```
[root@host01 ~]# radosgw-admin key create --uid=johndoe --key-type=s3 --gen-access-key --gen-secret
```

您还可以指定密钥和 secret。

要删除访问密钥，您需要指定用户和密钥：

1. 查找特定用户的访问密钥：

示例

```
[root@host01 ~]# radosgw-admin user info --uid=johndoe
```

access 键是输出中的 "access_key" 值：

示例

```
[root@host01 ~]# radosgw-admin user info --uid=johndoe
{
    "user_id": "johndoe",
    ...
    "keys": [
        {
            "user": "johndoe",
            "access_key": "0555b35654ad1656d804",
            "secret_key":
"h7GhxuBLTrlhVUyxSPUKUV8r/2EI4ngqJxD7iBdBYLhwluN30JaT3Q=="
        }
    ],
    ...
}
```

2. **指定用户 ID 和上一步中的访问密钥以删除访问密钥：**

语法

```
radosgw-admin key rm --uid=USER_ID --access-key ACCESS_KEY
```

示例

```
[root@host01 ~]# radosgw-admin key rm --uid=johndoe --access-key
0555b35654ad1656d804
```

8.4.12. 添加和删除管理功能

Ceph Storage 集群提供了一个管理 API，允许用户通过 REST API 运行管理功能。默认情况下，用户没有访问此 API 的权限。要让用户可以使用管理功能，请为用户提供管理功能。

要为用户添加管理功能，请运行以下命令：

语法

```
radosgw-admin caps add --uid=USER_ID--caps=CAPS
```

您可以为用户、存储桶、元数据和使用（使用）添加读取、写入或所有功能。

语法

```
--caps="[users|buckets|metadata|usage|zone]=[*|read|write|read, write]"
```

示例

```
[root@host01 ~]# radosgw-admin caps add --uid=johndoe --caps="users=*"
```

要从用户中删除管理功能，请运行以下命令：

示例

```
[root@host01 ~]# radosgw-admin caps remove --uid=johndoe --caps={caps}
```

8.5. 角色管理

作为存储管理员，您可以使用 radosgw-admin 命令创建、删除或更新角色以及与该角色关联的权限。

角色与用户类似，并附加了权限策略。它可以被任何身份假定。如果用户假定角色，则一组动态创建的临时凭据将返回给用户。角色可用于委派对用户、没有权限访问某些 S3 资源的应用程序和服务的访问权限。

8.5.1. 创建角色

使用 radosgw-admin role create 命令为用户创建一个角色。您需要在命令中创建具有 assume-role-policy-doc 参数的用户，这是授予授予角色权限的实体的信任关系策略文档。

先决条件

- 一个正在运行的 Red Hat Ceph Storage 集群。
- 安装 Ceph 对象网关.
- Ceph 对象网关节点的根级别访问权限.
- 已创建 S3 存储桶。

- 创建的用户具有访问权限的 S3 用户。

流程

- 创建角色：

语法

```
radosgw-admin role create --role-name=ROLE_NAME [--path=="PATH_TO_FILE"] [--assume-role-policy-doc=TRUST_RELATIONSHIP_POLICY_DOCUMENT]
```

示例

```
[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 的值默认为 /。

8.5.2. 获取角色

使用 get 命令获取有关角色的信息。

先决条件

- 一个正在运行的 Red Hat Ceph Storage 集群。
- 安装 Ceph 对象网关.
- Ceph 对象网关节点的根级别访问权限.
- 已创建 S3 存储桶。
- 已创建角色。
- 创建的用户具有访问权限的 S3 用户。

流程

- 获取有关角色的信息：

  语法

```
radosgw-admin role get --role-name=ROLE_NAME
```

  示例

```
[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\"]}]}"
}
```

其它资源

- 详情请参阅 **Red Hat Ceph Storage Object Gateway** 指南中的创建 角色 部分。

8.5.3. 列出角色

您可以使用 **list** 命令列出特定路径 中的角色。

先决条件

- 一个正在运行的 **Red Hat Ceph Storage** 集群。
- 安装 **Ceph** 对象网关.
- **Ceph** 对象网关节点的根级别访问权限.
- 已创建 **S3** 存储桶。
- 已创建角色。
- 创建的用户具有访问权限的 **S3** 用户。

流程

- *列出角色：*

语法

```
radosgw-admin role list
```

示例

```
[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. 更新角色的 assume 角色策略文档

您可以更新假定角色策略文档，该策略授予实体权限，以通过 modify 命令假定角色。

先决条件

- 一个正在运行的 Red Hat Ceph Storage 集群。
- 安装 Ceph 对象网关.
- Ceph 对象网关节点的根级别访问权限.
- 已创建 S3 存储桶。
- 已创建角色。
- 创建的用户具有访问权限的 S3 用户。

流程

- 修改角色的 assume 角色策略文档：

语法

```
radosgw-admin role-trust-policy modify --role-name=ROLE_NAME --assume-role-policy-doc=TRUST_RELATIONSHIP_POLICY_DOCUMENT
```

示例

```
[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. 获取附加到角色的权限策略

您可以使用 get 命令获取附加到角色的特定权限策略。

先决条件

- 一个正在运行的 Red Hat Ceph Storage 集群。
- 安装 Ceph 对象网关.
- Ceph 对象网关节点的根级别访问权限.
- 已创建 S3 存储桶。
- 已创建角色。
- 创建的用户具有访问权限的 S3 用户。

流程

- 获取权限策略：

  语法

```
radosgw-admin role-policy get --role-name=ROLE_NAME --policy-name=POLICY_NAME
```

示例

```
[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. 删除角色

您只能在删除附加的权限策略后删除角色。

先决条件

- 一个正在运行的 Red Hat Ceph Storage 集群。
- 安装 Ceph 对象网关.
- Ceph 对象网关节点的根级别访问权限.
- 已创建角色。
- 已创建 S3 存储桶。
- 创建的用户具有访问权限的 S3 用户。

流程

流程

1. **删除附加到角色的策略：**

语法

```
radosgw-admin role policy delete --role-name=ROLE_NAME --policy-name=POLICY_NAME
```

示例

```
[root@host01 ~]# radosgw-admin role policy delete --role-name=S3Access1 --policy-name=Policy1
```

2. **删除角色：**

语法

```
radosgw-admin role delete --role-name=ROLE_NAME
```

示例

```
[root@host01 ~]# radosgw-admin role delete --role-name=S3Access1
```

8.5.7. 更新附加到角色的策略

您可以使用 put 命令添加或更新附加到角色中的内联策略。

先决条件

- 一个正在运行的 Red Hat Ceph Storage 集群。
- 安装 Ceph 对象网关.
- Ceph 对象网关节点的根级别访问权限.
- 已创建 S3 存储桶。
- 已创建角色。
- 创建的用户具有访问权限的 S3 用户。

流程

- 更新内联策略：

语法

```
radosgw-admin role-policy put --role-name=ROLE_NAME --policy-name=POLICY_NAME --
policy-doc=PERMISSION_POLICY_DOCUMENT
```

示例

```
[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\"\}\]\}
```

在本例中，您可以将 Policy1 附加到角色 S3Access1 中，它允许对 example_bucket 的所有 S3 操作。

8.5.8. 列出附加到角色的权限策略

您可以使用 list 命令列出附加到角色的权限策略的名称。

先决条件

- 一个正在运行的 Red Hat Ceph Storage 集群。
- 安装 Ceph 对象网关.
- Ceph 对象网关节点的根级别访问权限.
- 已创建 S3 存储桶。
- 已创建角色。
- 创建的用户具有访问权限的 S3 用户。

流程

- 列出权限策略的名称：

语法

```
radosgw-admin role-policy list --role-name=ROLE_NAME
```

示例

```
[root@host01 ~]# radosgw-admin role-policy list --role-name=S3Access1

[
  "Policy1"
]
```

8.5.9. 删除附加到角色的策略

您可以使用 rm 命令删除附加到角色的权限策略。

先决条件

- *一个正在运行的 Red Hat Ceph Storage 集群。*
- *安装 Ceph 对象网关.*
- *Ceph 对象网关节点的根级别访问权限.*
- *已创建 S3 存储桶。*
- *已创建角色。*
- *创建的用户具有访问权限的 S3 用户。*

流程

- *删除权限策略：*

语法

```
radosgw-admin role policy delete --role-name=ROLE_NAME --policy-name=POLICY_NAME
```

示例

```
[root@host01 ~]# radosgw-admin role policy delete --role-name=S3Access1 --policy-name=Policy1
```

8.5.10. 更新角色的会话持续时间

您可以使用 update 命令更新角色的会话持续时间，以控制用户可以使用提供的凭证登录到帐户中的时间长度。

先决条件

- 一个正在运行的 Red Hat Ceph Storage 集群。
- 安装 Ceph 对象网关.
- Ceph 对象网关节点的根级别访问权限.
- 已创建 S3 存储桶。
- 已创建角色。
- 创建的用户具有访问权限的 S3 用户。

流程

- *使用 update 命令更新 max-session-duration ：*

语法

```
[root@node1 ~]# radosgw-admin role update --role-name=ROLE_NAME --max-session-duration=7200
```

示例

```
[root@node1 ~]# radosgw-admin role update --role-name=test-sts-role --max-session-duration=7200
```

验证

- *列出角色以验证更新：*

示例

```
[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\"]}]}"
    }
]
```

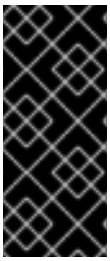
其它资源

- 详情请参阅 Red Hat Ceph Storage Developer Guide 中的 操作角色的 REST API 部分。

8.6. 配额管理

Ceph 对象网关允许您设置用户拥有的用户和 bucket 的配额。配额包括 bucket 中对象的最大数量，以及最大存储大小（以 MB 为单位）。

- Bucket： --bucket 选项允许您为用户拥有的存储桶指定配额。
- 最大对象： --max-objects 设置允许您指定对象的最大数量。负值将禁用此设置。
- 最大大小： --max-size 选项允许您为最大字节数指定配额。负值将禁用此设置。
- 配额范围： --quota-scope 选项设置配额的范围。选项为 bucket 和 user。bucket 配额应用到用户拥有的 bucket。用户配额应用到用户。



重要

具有大量对象的 bucket 可能会导致严重的性能问题。一个 bucket 中建议的最多对象数量为 100,000。要增加这个数量，请配置存储桶索引分片。详情请参阅 Configure bucket index resharding。

8.6.1. 设置用户配额

在启用配额前，您必须首先设置配额参数。

语法

```
radosgw-admin quota set --quota-scope=user --uid=USER_ID [--max-
objects=NUMBER_OF_OBJECTS] [--max-size=MAXIMUM_SIZE_IN_BYTES]
```

-

示例

```
[root@host01 ~]# radosgw-admin quota set --quota-scope=user --uid=johndoe --max-objects=1024 --max-size=1024
```

num *对象和* / *或* **max size** *的负值表示禁用特定的配额属性检查。*

8.6.2. 启用和禁用用户配额

设置用户配额后，您可以启用它。

语法

```
radosgw-admin quota enable --quota-scope=user --uid=USER_ID
```

您可以禁用启用的用户配额。

语法

```
radosgw-admin quota disable --quota-scope=user --uid=USER_ID
```

8.6.3. 设置存储桶配额

bucket 配额应用到指定 uid 拥有的 bucket。它们独立于用户。

语法

```
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 的负值表示禁用特定的配额属性检查。

8.6.4. 启用和禁用存储桶配额

设置存储桶配额后，您可以启用它。

语法

```
radosgw-admin quota enable --quota-scope=bucket --uid=USER_ID
```

您可以禁用启用的存储桶配额。

语法

```
radosgw-admin quota disable --quota-scope=bucket --uid=USER_ID
```

8.6.5. 获取配额设置

您可以通过用户信息 API 访问每个用户的配额设置。要使用 CLI 接口读取用户配额设置信息，请运行以下命令：

语法

```
radosgw-admin user info --uid=USER_ID
```

要获得租户用户的配额设置，请指定用户 ID 和租户名称：

语法

```
radosgw-admin user info --uid=USER_ID --tenant=TENANT
```

8.6.6. 更新配额统计

配额统计异步更新。您可以手动更新所有用户和所有存储桶的配额统计，以检索最新的配额统计。

语法

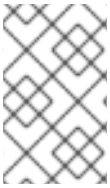
```
radosgw-admin user stats --uid=USER_ID --sync-stats
```

8.6.7. 获取用户配额使用量统计

要查看用户消耗的配额量，请运行以下命令：

语法

```
radosgw-admin user stats --uid=USER_ID
```



注意

您应当使用 **--sync-stats** 选项执行 **radosgw-admin user stats**，以接收最新的数据。

8.6.8. 配额缓存

为每个 **Ceph** 网关实例缓存配额统计数据。如果有多个实例，缓存可以防止完全强制执行配额，因为每个实例对配额有不同的视图。控制此功能的选项是 **rgw bucket** 配额 **ttl**、**rgw** 用户配额 **bucket** 同步间隔，以及 **rgw** 用户配额同步间隔。这些值越大，配额操作效率越高，但多个实例不同步就会越高。这些值越低，将更接近于完美地实施多个实例。如果所有三个都是 **0**，则有效禁用配额缓存，多个实例也具有完美的配额实施。

8.6.9. 读取和写入全局配额

您可以在 **zonegroup map** 中读取和写入配额设置。获取 **zonegroup map**：

```
[root@host01 ~]# radosgw-admin global quota get
```

全局配额设置可以使用 **global quota** 的与 **quota set, quota enable, 和 quota disable** 命令相关的部分，例如：

```
[root@host01 ~]# radosgw-admin global quota set --quota-scope bucket --max-objects 1024
[root@host01 ~]# radosgw-admin global quota enable --quota-scope bucket
```

注意

在存在 **realm** 和 **period** 的多站点配置中，必须使用 **period update --commit** 提交对全局配额的更改。如果没有 **period**，则必须重启 **Ceph** 对象网关，才能使更改生效。

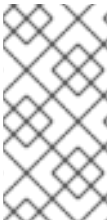
8.7. BUCKET 管理

作为存储管理员，在使用 **Ceph** 对象网关时，您可以通过在用户之间移动 **bucket** 并将它们重命名来管

理存储桶。您可以创建存储桶通知，在特定事件时触发。此外，您可以在 Ceph 对象网关中发现在存储集群的生命周期内可能会发生孤立或泄漏的对象。

注意

当数百万对象上传到具有高容量率的 Ceph 对象网关 bucket 时，会使用 radosgw-admin bucket stats 命令报告不正确的 num_objects。使用 radosgw-admin bucket list 命令，您可以更正 num_objects 参数的值。



注意

在多站点集群中，从次要站点中删除存储桶不会与主站点同步元数据更改。因此，红帽建议只从主站点中删除存储桶，而不是从次要站点中删除存储桶。

8.7.1. 重命名存储桶

您可以重命名存储桶。如果要在存储桶名称中允许下划线，请将 rgw_relaxed_s3_bucket_names 选项设置为 true。

先决条件

- 一个正在运行的 Red Hat Ceph Storage 集群。
- 安装 Ceph 对象网关软件.
- 一个现有存储桶。

流程

1. 列出存储桶：

示例

```
[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. ***重命名存储桶：***

语法

```
radosgw-admin bucket link --bucket=ORIGINAL_NAME --bucket-new-name=NEW_NAME --uid=USER_ID
```

示例

```
[ceph: root@host01 /]# radosgw-admin bucket link --bucket=s3bucket1 --bucket-new-name=s3newb --uid=testuser
```

如果存储桶位于租户中，还要指定租户：

语法

```
radosgw-admin bucket link --bucket=tenant/ORIGINAL_NAME --bucket-new-name=NEW_NAME --uid=TENANT$USER_ID
```

示例

```
[ceph: root@host01 /]# radosgw-admin bucket link --bucket=test/s3bucket1 --bucket-new-name=s3newb --uid=test$testuser
```

3. 验证存储桶已被重命名：

示例

```
[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. 删除存储桶

使用 Ceph 对象网关配置从 Red Hat Ceph Storage 集群中删除存储桶。

当存储桶没有对象时，您可以运行 **radosgw-admin bucket rm** 命令。如果存储桶中存在对象，您可以使用 **--purge-objects** 选项。

对于多站点配置，红帽建议从主站点中删除存储桶。

先决条件

- *一个正在运行的 Red Hat Ceph Storage 集群。*
- *安装 Ceph 对象网关软件.*
- *一个现有存储桶。*

流程

1. *列出存储桶。*

   示例

```
[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. *移除存储桶。*

   语法

```
radosgw-admin bucket rm --bucket=BUCKET_NAME
```

示例

```
[ceph: root@host01 /]# radosgw-admin bucket rm --bucket=s3bucket1
```

3. **如果存储桶有对象，请运行以下命令：**

语法

```
radosgw-admin bucket rm --bucket=BUCKET --purge-objects --bypass-gc
```

示例

```
[ceph: root@host01 /]# radosgw-admin bucket rm --bucket=s3bucket1 --purge-objects --
bypass-gc
```

--purge-objects 选项**清除对象，--bypass-gc** 选项**会触发删除对象，而无需垃圾收集器使进程更高效。**

4. **验证存储桶已被删除。**

示例

```
[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. 移动存储桶

radosgw-admin bucket 实用程序提供在用户之间移动 **bucket** 的功能。为此，请将存储桶链接到新用户，并将 **bucket** 的所有权更改为新用户。

您可以移动存储桶：

- 两个非租户用户之间
- 两个租户用户间
- 在非租户用户到租户用户间

先决条件

- 一个正在运行的 **Red Hat Ceph Storage** 集群。
- 已安装 **Ceph** 对象网关。
- **S3** 存储桶。

- 各种租户和非租户用户。

8.7.3.1. 在非租户用户之间移动存储桶

radosgw-admin bucket chown 命令提供将 **bucket** 的所有权及其包含的所有对象从一个用户更改为另一个用户的功能。为此，请从当前用户取消链接存储桶，将它链接到新用户，然后将 **bucket** 的所有权更改为新用户。

流程

1. 将存储桶链接到一个新用户：

语法

```
radosgw-admin bucket link --uid=USER --bucket=BUCKET
```

示例

```
[ceph: root@host01 /]# radosgw-admin bucket link --uid=user2 --bucket=data
```

2. 验证存储桶已成功链接到 **user2** ：

示例

```
[ceph: root@host01 /]# radosgw-admin bucket list --uid=user2
[
    "data"
]
```

3. 将存储桶的所有权更改为新用户：

语法

```
radosgw-admin bucket chown --uid=user --bucket=bucket
```

示例

```
[ceph: root@host01 /]# radosgw-admin bucket chown --uid=user2 --bucket=data
```

4. 通过检查以下命令输出中的 **owner** 行来验证 **data** 存储桶的所有权是否已成功更改：

示例

```
[ceph: root@host01 /]# radosgw-admin bucket list --bucket=data
```

8.7.3.2. 在租户用户之间移动存储桶

您可以在租户用户和另一个租户用户之间移动存储桶。

流程

1. 将存储桶链接到一个新用户：

语法

```
radosgw-admin bucket link --bucket=CURRENT_TENANT/BUCKET --
uid=NEW_TENANT$USER
```

示例

```
[ceph: root@host01 /]# radosgw-admin bucket link --bucket=test/data --uid=test2$user2
```

2. *验证存储桶已成功链接到 **user2** ：*

```
[ceph: root@host01 /]# radosgw-admin bucket list --uid=test$user2
[
    "data"
]
```

3. ***将存储桶的所有权更改为新用户：***

语法

```
radosgw-admin bucket chown --bucket=NEW_TENANT/BUCKET --
uid=NEW_TENANT$USER
```

示例

```
[ceph: root@host01 /]# radosgw-admin bucket chown --bucket='test2/data' --uid='test$tuser2'
```

4. 通过检查以下命令输出中的 **owner** 行来验证 **data** 存储桶的所有权是否已成功更改：

```
[ceph: root@host01 /]# radosgw-admin bucket list --bucket=test2/data
```

8.7.3.3. 将存储桶从非租户用户移到租户的用户

您可以将存储桶从非租户用户移到租户用户。

流程

1. 可选：如果您还没有多个租户，您可以通过启用 **rgw_keystone_implicit_tenants** 并从外部租户访问 **Ceph** 对象网关来创建它们：

   启用 **rgw_keystone_implicit_tenants** 选项：

   示例

```
[ceph: root@host01 /]# ceph config set client.rgw rgw_keystone_implicit_tenants true
```

   使用 **s3cmd** 或 **swift** 命令从外部租户访问 **Ceph** 对象网关：

   示例

```
[ceph: root@host01 /]# swift list
```

   或使用 **s3cmd:**

示例

```
[ceph: root@host01 /]# s3cmd ls
```

从外部租户进行第一次访问可创建等效的 Ceph 对象网关用户。

2. 将存储桶移到租户的用户：

语法

```
radosgw-admin bucket link --bucket=/BUCKET --uid='TENANT$USER'
```

示例

```
[ceph: root@host01 /]# radosgw-admin bucket link --bucket=/data --uid='test$tenanted-user'
```

3. 验证 **data** 存储桶是否已成功链接到 **tenanted-user** ：

示例

```
[ceph: root@host01 /]# radosgw-admin bucket list --uid='test$tenanted-user'
[
    "data"
]
```

4. 将存储桶的所有权更改为新用户：

语法

```
radosgw-admin bucket chown --bucket='tenant/bucket name' --uid='tenant$user'
```

示例

```
[ceph: root@host01 /]# radosgw-admin bucket chown --bucket='test/data' --
uid='test$tenanted-user'
```

5. 通过检查以下命令输出中的 **owner** 行来验证 **data** 存储桶的所有权是否已成功更改：

示例

```
[ceph: root@host01 /]# radosgw-admin bucket list --bucket=test/data
```

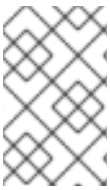
8.7.3.4. 查找孤立和泄漏对象

健康的存储集群没有任何孤立或泄漏的对象，但在某些情况下可能会发生孤立或泄漏的对象。

存储集群中存在孤立对象，并且具有与 **RADOS** 对象关联的对象 **ID**。但是，在 **bucket** 索引引用中，没有通过 **S3** 对象引用 **RADOS** 对象。例如，如果 **Ceph** 对象网关在操作的中间发生，这可能会导致一些

对象变得孤立。另外，未发现的错误可能导致孤立对象发生。

您可以了解 Ceph 对象网关对象如何映射到 RADOS 对象。radosgw-admin 命令提供了一个用于搜索和生成这些潜在孤立或泄漏对象的列表的工具。使用 radoslist 子命令可显示 bucket 中存储的对象，或者存储集群中的所有 bucket。rgw-orphan-list 脚本显示池中的孤立对象。



注意

radoslist 子命令将替代已弃用的 orphans find 和 orphans finish 子命令。

重要

不要使用无索引存储桶， 因为所有对象都显示为 孤立。

身份孤立对象的另一种方法是运行 rados -p <pool> ls | grep BUCKET_ID 命令。

先决条件

- 一个正在运行的 Red Hat Ceph Storage 集群。
- 正在运行的 Ceph 对象网关.

流程

1. 生成存储桶中保存数据的对象列表。

语法

```
radosgw-admin bucket radoslist --bucket BUCKET_NAME
```

示例

```
[root@host01 ~]# radosgw-admin bucket radoslist --bucket mybucket
```

注意

如果省略 **BUCKET_NAME**，则会显示所有存储桶中的所有对象。

2. 检查 **rgw-orphan-list** 的版本。

示例

```
[root@host01 ~]# head /usr/bin/rgw-orphan-list
```

版本应该是 **2023-01-11** 或更新版本。

3. 创建一个需要生成孤立项列表的目录。

示例

```
[root@host01 ~]# mkdir orphans
```

4. 导航到之前创建的目录。

示例

```
[root@host01 ~]# cd orphans
```

5. **从池列表中，选择要在其中查找孤立的池。根据集群中的对象，此脚本可能会长时间运行。**

示例

```
[root@host01 orphans]# rgw-orphan-list
```

示例

```
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?
```

输入池名称以搜索孤立项。

重要

在使用 rgw-orphan-list 命令而非元数据池时，必须指定数据池。

6. 查看 **rgw-orphan-list** 工具用法的详细信息。

Synatx

```
rgw-orphan-list -h
rgw-orphan-list POOL_NAME /DIRECTORY
```

示例

```
[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. 运行 **ls -l** 命令，以验证以 **error** 结尾的文件应该为零长度，表示脚本运行没有任何问题。

示例

```
[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. *检查列出的孤立对象。*

示例

```
[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. *删除孤立对象：*

语法

```
rados -p POOL_NAME rm OBJECT_NAME
```

示例

```
[root@host01 orphans]# rados -p default.rgw.buckets.data rm myobject
```

警告

验证您是否删除了正确的对象。运行 **rados rm** 命令从存储集群中移除数据。

8.7.3.5. 管理存储桶索引条目

您可以使用 **radosgw-admin bucket check** 子命令，在 Red Hat Ceph Storage 集群中管理 Ceph 对象网关的存储桶索引条目。

与多部分上传对象相关的每个 **bucket** 索引条目都与其对应的 **.meta** 索引条目匹配。对于给定多部分上传，应该有一个 **.meta** 条目。如果某个部分找不到对应的 **.meta** 条目，它会在输出的部分中列出"孤立"片段条目。

bucket 的统计信息存储在存储桶索引标头中。此阶段加载这些标头，并迭代存储桶索引中的所有普通对象条目，并重新计算统计。然后，它会在标有 **"existing_header" 和 "calculated_header"** 的部分中分别显示实际和计算的统计数据，以便可以比较它们。

如果您将 **--fix** 选项与 **bucket check** 子命令搭配使用，它会从存储桶索引中删除"孤立"条目，并用它计算的标头中覆盖标头中的现有统计。这会导致输出中列出所有条目，包括版本控制中使用的多个条目。

先决条件

- 一个正在运行的 Red Hat Ceph Storage 集群。
- 正在运行的 Ceph 对象网关.
- 新创建的存储桶。

流程

1. **检查特定存储桶的存储桶索引：**

语法

```
radosgw-admin bucket check --bucket=BUCKET_NAME
```

示例

```
[root@rgw ~]# radosgw-admin bucket check --bucket=mybucket
```

2. **修复存储桶索引中的不一致问题，包括删除孤立对象：**

语法

```
radosgw-admin bucket check --fix --bucket=BUCKET_NAME
```

示例

```
[root@rgw ~]# radosgw-admin bucket check --fix --bucket=mybucket
```

8.7.3.6. bucket 通知

bucket 通知提供了一种方式，可以在 bucket 中发生特定事件时从 Ceph 对象网关发送信息。bucket 通知可以发送到 HTTP、AMQP0.9.1 和 Kafka 端点。必须创建一个通知条目，以便为特定存储桶上的事件和特定主题发送存储桶通知。可以在事件类型的子集上创建 bucket 通知，也可以默认为所有事件类型创建 bucket 通知。bucket 通知可以根据密钥前缀或后缀、匹配键的正则表达式、附加到对象或对象标签的元数据属性过滤出事件。bucket 通知具有 REST API，用于为 bucket 通知机制提供配置和控制接口。

注意

bucket 通知 API 会被默认启用。如果明确设置了 rgw_enable_apis 配置参数，请确保包含 s3 和 notifications。要进行验证，请运行 ceph --admin-daemon /var/run/ceph/ceph-client.rgw.NAME.asok config get rgw_enable_apis 命令。将 NAME 替换为 Ceph 对象网关实例名称。

使用 CLI 的主题管理

您可以管理 Ceph 对象网关存储桶的列表、获取和删除主题：

- 列出主题 ：运行以下命令列出所有主题的配置：

  示例

  ```
  [ceph: host01 /]# radosgw-admin topic list
  ```

- 获取主题 ：运行以下命令以获取特定主题的配置：

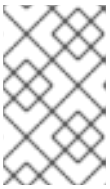
  示例

  ```
  [ceph: host01 /]# radosgw-admin topic get --topic=topic1
  ```

- 删除主题 ：运行以下命令以删除特定主题的配置：

示例

```
[ceph: host01 /]# radosgw-admin topic rm --topic=topic1
```



注意

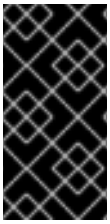
即使 Ceph 对象网关存储桶已配置至该主题，该主题也会被删除。

8.7.3.7. 创建存储桶通知

在 bucket 级别上创建 bucket 通知。通知配置具有 Red Hat Ceph Storage Object Gateway S3 事件、ObjectCreated、ObjectRemoved 和 ObjectLifecycle:Expiration。它们需要与目的地一起发布，以发送存储桶通知。bucket 通知是 S3 操作。

先决条件

- *一个正在运行的 Red Hat Ceph Storage 集群。*
- *正在运行的 HTTP 服务器、RabbitMQ 服务器或 Kafka 服务器。*
- *根级别访问权限.*
- *安装 Red Hat Ceph Storage 存储对象网关.*
- *用户 access key 和 secret key。*
- *端点参数.*



重要

红帽支持 ObjectCreate 事件，如 put, post, multipartUpload, 和 copy。红帽还支持 ObjectRemove 事件，如 object_delete 和 s3_multi_object_delete。

流程

1. 创建 S3 存储桶。

2. 为 http、amqp 或 kafka 协议创建一个 SNS 主题。

3. 为 s3:objectCreate,s3:objectRemove, 和 s3:ObjectLifecycle:Expiration 事件创建一个 S3 存储桶通知：

   示例

```
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. 在存储桶中创建 S3 对象。

5. 在 http、rabbitmq 或 kafka 接收器验证对象创建事件。

6. 删除对象。

7. 在 http、rabbitmq 或 kafka 接收器验证对象删除事件。

8.7.4. S3 bucket 复制 API

S3 bucket 复制 API 已被实施，允许用户在不同 bucket 之间创建复制规则。请注意，虽然 AWS 复制功能允许在同一区内复制存储桶，但 Ceph 对象网关目前不允许它。但是，Ceph 对象网关 API 也添加了一个 Zone 数组，允许用户选择要同步特定 bucket 的区域。

8.7.4.1. 创建 S3 存储桶复制

为存储桶创建复制配置或替换现有配置。

复制配置必须至少包含一个规则。每个规则通过过滤源存储桶中的对象来识别要复制的对象子集。

先决条件

- 一个正在运行的 Red Hat Ceph Storage 集群。
- 已创建 zone group 级别策略。

流程

1. 创建包含复制详情的复制配置文件：

语法

```
{
"Role": "arn:aws:iam::account-id:role/role-name",
   "Rules": [
      {
         "ID": "String",
         "Status": "Enabled",
         "Priority": 1,
         "DeleteMarkerReplication": { "Status": "Enabled"|"Disabled" },
         "Destination": {
            "Bucket": "BUCKET_NAME"
         }
```

```
        }
    ]
}
```

示例

```
[root@host01 ~]# cat replication.json
{
    "Role": "arn:aws:iam::account-id:role/role-name",
    "Rules": [
        {
            "ID": "pipe-bkt",
            "Status": "Enabled",
            "Priority": 1,
            "DeleteMarkerReplication": { "Status": "Disabled" },
            "Destination": {
                "Bucket": "testbucket"
            }
        }
    ]
}
```

2. **创建 S3 API 放置存储桶复制：**

语法

```
aws --endpoint-url=RADOSGW_ENDPOINT_URL s3api put-bucket-replication --bucket
BUCKET_NAME --replication-configuration file://REPLICATION_CONFIIRATION_FILE.json
```

示例

```
[root@host01 ~]# aws --endpoint-url=http://host01:80 s3api put-bucket-replication --bucket
testbucket --replication-configuration file://replication.json
```

验证

1. ***使用 sync policy get 命令验证同步策略。***

语法

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

注意

应用复制策略时，规则将转换为 sync-policy 规则，称为 管道，并归类为启用和禁用 。

- ***启用 ：这些管道已启用，并且组状态设置为 'rgw_sync_policy_group:STATUS'。例如， s3-bucket-replication:enabled。***
- ***disabled ：此集合下的管道不是活跃的，组状态被设置为 'rgw_sync_policy_group:STATUS'。例如， s3-bucket-replication:disabled。***

由于可能存在多个规则，它们可以配置为复制策略的一部分，它有两个独立的组（一个带有 'enabled'，另一个带有"allowed"状态）来准确映射每个组。

示例

```
[ceph: root@host01 /]# radosgw-admin sync policy get --bucket testbucket
{
    "groups": [
        {
            "id": "s3-bucket-replication:disabled",
```

```
            "data_flow": {},
            "pipes": [],
            "status": "allowed"
        },
        {
            "id": "s3-bucket-replication:enabled",
            "data_flow": {},
            "pipes": [
                {
                    "id": "",
                    "source": {
                        "bucket": "*",
                        "zones": [
                            "*"
                        ]
                    },
                    "dest": {
                        "bucket": "testbucket",
                        "zones": [
                            "*"
                        ]
                    },
                    "params": {
                        "source": {},
                        "dest": {},
                        "priority": 1,
                        "mode": "user",
                        "user": "s3cmd"
                    }
                }
            ],
            "status": "enabled"
        }
    ]
}
```

其它资源

- 详情请参阅 **Red Hat Ceph Storage Object Gateway** 指南中的使用多站点同步策略部分。 https://access.redhat.com/documentation/zh-cn/red_hat_ceph_storage/7/html-single/object_gateway_guide/#using-multisite-sync-policies

8.7.4.2. 获取 S3 存储桶复制

您可以检索存储桶的复制配置。

先决条件

- *一个正在运行的 Red Hat Ceph Storage 集群。*
- *已创建 zone group 级别策略。*
- *创建的 S3 存储桶复制。*

流程

- *获取 S3 API 放置存储桶复制：*

  语法

```
aws s3api get-bucket-replication --bucket BUCKET_NAME --endpoint-
url=RADOSGW_ENDPOINT_URL
```

  示例

```
[root@host01 ~]# aws s3api get-bucket-replication --bucket testbucket --endpoint-
url=http://host01:80
{

"ReplicationConfiguration": {
    "Role": "",
    "Rules": [
        {
            "ID": "pipe-bkt",
            "Status": "Enabled",
            "Priority": 1,
            "Destination": {
                Bucket": "testbucket"
            }
        }
    ]
  }

}
```

8.7.4.3. 删除 S3 存储桶复制

从存储桶中删除复制配置。

bucket 所有者可以为其他人授予权限来删除复制配置。

先决条件

- 一个正在运行的 Red Hat Ceph Storage 集群。
- 已创建 zone group 级别策略。
- 创建的 S3 存储桶复制。

流程

1. 删除 S3 API 放置存储桶复制：

语法

```
aws s3api delete-bucket-replication --bucket BUCKET_NAME --endpoint-
url=RADOSGW_ENDPOINT_URL
```

示例

```
[root@host01 ~]# aws s3api delete-bucket-replication --bucket testbucket --endpoint-
url=http://host01:80
```

验证

- *验证现有的复制规则是否已删除：*

  语法

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

  示例

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

8.7.4.4. 为用户禁用 S3 存储桶复制

作为管理员，您可以为其他用户设置用户策略，以限制对位于该特定用户/用户的存储桶执行任何 s3 复制 API 操作。

先决条件

- *一个正在运行的 Red Hat Ceph Storage 集群。*
- *已创建 zonegroup-level 策略。*

流程

1. *创建用户策略配置文件以拒绝对 S3 存储桶复制 API 的访问：*

示例

```
[root@host01 ~]# cat user_policy.json
{
 "Version":"2012-10-17",
 "Statement":
 {
  "Effect":"Deny",
  "Action":
  [
   "s3:PutReplicationConfiguration",
   "s3:GetReplicationConfiguration",
   "s3:DeleteReplicationConfiguration"
  ],
  "Resource": "arn:aws:s3:::*",
 }
}
```

2. **以 admin 用户身份，将用户策略设置为 user 以禁用用户对 S3 API 的访问：**

语法

```
aws --endpoint-url=ENDPOINT_URL iam put-user-policy --user-name USER_NAME --
policy-name USER_POLICY_NAME --policy-document POLICY_DOCUMENT_PATH
```

示例

```
[root@host01 ~]# aws --endpoint-url=http://host01:80 iam put-user-policy --user-name
newuser1 --policy-name userpolicy --policy-document file://user_policy.json
```

验证

- 以 admin 用户身份，验证用户策略集：

语法

```
aws --endpoint-url=ENDPOINT_URL iam get-user-policy --user-name USER_NAME --policy-name USER_POLICY_NAME --region us
```

示例

```
[root@host01 ~]# aws --endpoint-url=http://host01:80 iam get-user-policy --user-name newuser1 --policy-name userpolicy --region us
```

- 作为 admin 用户设置了用户策略的用户，请尝试在 S3 存储桶复制 API 操作下执行，以验证操作是否如预期拒绝。
  - 创建 S3 存储桶复制
  - 获取 S3 存储桶复制
  - 删除 S3 存储桶复制

其它资源

- 详情请参阅 Red Hat Ceph Storage Object Gateway Guide 中的 S3 bucket 复制 API 部分。

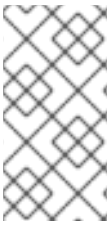
其它资源

- 如需更多信息，请参阅 Red Hat Ceph Storage 开发人员指南。

8.8. BUCKET 生命周期

作为存储管理员，您可以使用存储桶生命周期配置来管理对象，以便它们在整个生命周期中有效地存储。例如，您可以根据您的用例，将对象转换到更便宜的存储类、归档甚至删除它们。

RADOS 网关支持 S3 API 对象到期，方法是利用为一组 bucket 对象定义规则。每个规则都有一个前缀，用于选择对象，以及对象不可用的天数。



注意

radosgw-admin lc reshard 命令在 Red Hat Ceph Storage 3.3 中已弃用，Red Hat Ceph Storage 4 及更新的版本中不被支持。

8.8.1. 创建生命周期管理策略

您可以使用标准 S3 操作来管理存储桶生命周期策略配置，而不是使用 radosgw-admin 命令。RADOS 网关仅支持应用于 bucket 的 Amazon S3 API 策略语言的子集。生命周期配置包含为一组 bucket 对象定义的一个或多个规则。

先决条件

- 正在运行的红帽存储集群。
- 安装 Ceph 对象网关.
- Ceph 对象网关节点的根级别访问权限.
- 已创建 S3 存储桶。
- 创建的用户具有访问权限的 S3 用户。

- 使用安装的 AWS CLI 软件包访问 Ceph 对象网关客户端。

流程

1. 为生命周期配置创建 JSON 文件：

示例

```
[user@client ~]$ vi lifecycle.json
```

2. 在文件中添加特定的生命周期配置规则：

示例

```
{
 "Rules": [
       {
      "Filter": {
       "Prefix": "images/"
      },
      "Status": "Enabled",
      "Expiration": {
       "Days": 1
      },
      "ID": "ImageExpiration"
     }
    ]
}
```

生命周期配置示例在 1 天后在 images 目录中过期对象。

3. 在存储桶上设置生命周期配置：

语法

```
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
```

示例

```
[user@client ~]$ aws --endpoint-url=http://host01:80 s3api put-bucket-lifecycle-configuration --bucket testbucket --lifecycle-configuration file://lifecycle.json
```

*在本例中，**lifecycle.json** 文件存在于当前目录中。*

验证

- *检索存储桶的生命周期配置：*

语法

```
aws --endpoint-url=RADOSGW_ENDPOINT_URL:PORT s3api get-bucket-lifecycle-configuration --bucket BUCKET_NAME
```

示例

```
[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"
     }
    ]
}
```

- ***可选：在 Ceph 对象网关节点中，登录到 Cephadm shell 并检索存储桶生命周期配置：***

语法

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

示例

```
[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
   }
  }
   ]
}
```

其它资源

- 详情请参阅 **Red Hat Ceph Storage Developer Guide** 中的 **S3** 存储桶生命周期 部分。
- 有关使用 **AWS CLI** 管理生命周期配置的更多信息，请参阅 **Amazon Simple Storage Service** 文档中的在存储桶上 设置生命周期配置 部分。

8.8.2. 删除生命周期管理策略

您可以使用 **s3api delete-bucket-lifecycle** 命令删除指定存储桶的生命周期管理策略。

先决条件

- 正在运行的红帽存储集群。
- 安装 **Ceph** 对象网关.

- *Ceph 对象网关节点的根级别访问权限.*
- *已创建 S3 存储桶。*
- *创建的用户具有访问权限的 S3 用户。*
- *使用安装的 AWS CLI 软件包访问 Ceph 对象网关客户端。*

流程

- *删除生命周期配置：*

  语法

  ```
  aws --endpoint-url=RADOSGW_ENDPOINT_URL:PORT s3api delete-bucket-lifecycle --bucket BUCKET_NAME
  ```

  示例

  ```
  [user@client ~]$ aws --endpoint-url=http://host01:80 s3api delete-bucket-lifecycle --bucket testbucket
  ```

验证

- *检索存储桶的生命周期配置：*

  语法

```
aws --endpoint-url=RADOSGW_ENDPOINT_URL:PORT s3api get-bucket-lifecycle-
configuration --bucket BUCKET_NAME
```

示例

```
[user@client ~]# aws --endpoint-url=http://host01:80  s3api get-bucket-lifecycle-configuration
--bucket testbucket
```

- **可选：在 Ceph 对象网关节点中，检索存储桶生命周期配置：**

语法

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

示例

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

注意

如果存储桶生命周期策略不存在，命令不会返回任何信息。

其它资源

- 详情请参阅 Red Hat Ceph Storage Developer Guide 中的 S3 存储桶生命周期 部分。

8.8.3. 更新生命周期管理策略

您可以使用 s3cmd put-bucket-lifecycle-configuration 命令更新生命周期管理策略。

注意

put-bucket-lifecycle-configuration 覆盖现有的存储桶生命周期配置。如果要保留任何当前生命周期策略设置，则必须将它们包含在生命周期配置文件中。

先决条件

- 正在运行的红帽存储集群。
- 安装 Ceph 对象网关.
- Ceph 对象网关节点的根级别访问权限.
- 已创建 S3 存储桶。
- 创建的用户具有访问权限的 S3 用户。
- 使用安装的 AWS CLI 软件包访问 Ceph 对象网关客户端。

流程

1. 为生命周期配置创建 JSON 文件：

示例

```
[user@client ~]$ vi lifecycle.json
```

2. *在文件中添加特定的生命周期配置规则：*

示例

```
{
 "Rules": [
       {
      "Filter": {
       "Prefix": "images/"
      },
      "Status": "Enabled",
      "Expiration": {
       "Days": 1
      },
      "ID": "ImageExpiration"
     },
  {
   "Filter": {
    "Prefix": "docs/"
   },
   "Status": "Enabled",
   "Expiration": {
    "Days": 30
   },
   "ID": "DocsExpiration"
  }
 ]
}
```

3. *更新存储桶上的生命周期配置：*

语法

```
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_FIL
E.json
```

示例

```
[user@client ~]$ aws --endpoint-url=http://host01:80 s3api put-bucket-lifecycle-configuration
--bucket testbucket --lifecycle-configuration file://lifecycle.json
```

验证

- **检索存储桶的生命周期配置：**

语法

```
aws --endpointurl=RADOSGW_ENDPOINT_URL:PORT s3api get-bucket-lifecycle-
configuration --bucket BUCKET_NAME
```

示例

```
[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"
    }
  ]
}
```

- ***可选：在 Ceph 对象网关节点中，登录到 Cephadm shell 并检索存储桶生命周期配置：***

语法

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

示例

```
[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
  }
```

```
    }
  ]
}
```

其它资源

- 如需了解有关 Amazon S3 存储桶生命周期 的详细信息，请参阅 Red Hat Ceph Storage 开发人员指南。

8.8.4. 监控存储桶生命周期

您可以使用 radosgw-admin lc list 和 radosgw-admin lc process 命令监控生命周期并手动处理存储桶的生命周期。

先决条件

- 一个正在运行的 Red Hat Ceph Storage 集群。
- Ceph 对象网关节点的根级别访问权限.
- 创建应用生命周期配置策略的 S3 存储桶。

流程

1. 登录到 Cephadm shell：

   示例

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

2. *列出存储桶生命周期进度：*

示例

```
[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”
  }
]
```

bucket 生命周期处理状态可以是以下之一：

- *UNINITIAL - 进程尚未运行。*
- *PROCESSING - 进程当前正在运行。*
- *COMPLETE - 进程已完成。*

3. *可选：您可以手动处理存储桶生命周期策略：*

a. *处理单个存储桶的生命周期策略：*

语法

```
radosgw-admin lc process --bucket=BUCKET_NAME
```

示例

```
[ceph: root@host01 /]# radosgw-admin lc process --bucket=testbucket1
```

b. **立即处理所有存储桶生命周期策略：**

示例

```
[ceph: root@host01 /]# radosgw-admin lc process
```

验证

- **列出存储桶生命周期策略：**

```
[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”
    }
]
```

其它资源

- 详情请参阅 Red Hat Ceph Storage Developer Guide 中的 S3 存储桶生命周期 部分。

8.8.5. 配置生命周期过期窗口

您可以通过设置 rgw_lifecycle_work_time 参数，设置生命周期管理进程每天运行的时间。默认情况下，生命周期处理每天都会进行一次。

先决条件

- 一个正在运行的 Red Hat Ceph Storage 集群。
- 安装 Ceph 对象网关.
- Ceph 对象网关节点的根级别访问权限.

流程

1. 登录到 Cephadm shell：

   示例

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

2. 设置生命周期过期时间：

   语法

   ```
   ceph config set client.rgw rgw_lifecycle_work_time %D:%D-%D:%D
   ```

将 %d:%d-%d:%d 替换为 start_hour:start_minute-end_hour:end_minute。

示例

```
[ceph: root@host01 /]# ceph config set client.rgw rgw_lifecycle_work_time 06:00-08:00
```

验证

- 检索生命周期到期工作时间：

示例

```
[ceph: root@host01 /]# ceph config get client.rgw rgw_lifecycle_work_time

06:00-08:00
```

其它资源

- 详情请参阅 Red Hat Ceph Storage Developer Guide 中的 S3 存储桶生命周期 部分。

8.8.6. S3 bucket 生命周期在存储集群中转换

您可以使用 bucket 生命周期配置来管理对象，以便在整个对象生命周期内有效存储对象。对象生命周期转换规则允许您在对象生命周期内管理并有效地存储对象。您可以将对象转换到更便宜的存储类、存档甚至删除它们。

您可以为以下对象创建存储类：

- 快速介质，如 SSD 或 NVMe 用于 I/O 敏感工作负载。
- 速度较慢的磁介质，如 SAS 或 SATA 以进行归档。

您可以为热存储类和冷存储类之间的数据移动创建调度。您可以在指定时间后调度此移动，以便对象过期并被永久删除，例如，您可以在创建对象 30 天后将其转换为存储类，甚至可以在创建对象后一年将对象归档到存储类。您可以通过转换规则完成此操作。这个规则适用于从一个存储类转换到另一个存储类的对象。生命周期配置包含使用 <Rule> 元素的一个或多个规则。

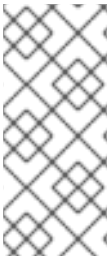
其它资源

- 如需了解有关 bucket 生命周期 的详细信息，请参阅 Red Hat Ceph Storage 开发人员指南。

8.8.7. 将对象从一个存储类转换到另一个存储类

对象生命周期转换规则允许您将对象从一个存储类转换为另一个类。

您可以在复制池、纠删代码池之间迁移数据，复制到纠删代码池，或使用 Ceph 对象网关生命周期转换策略迁移到复制池。



注意

在多站点配置中，当第一个站点应用生命周期转换规则时，将对象从一个数据池转换到同一存储集群中的另一个数据池，则同一规则对第二个站点有效，如果第二个站点具有通过 rgw 应用创建并启用相应的数据池。

先决条件

- 安装 Ceph 对象网关软件.
- Ceph 对象网关节点的根级别访问权限.
- 创建的用户具有访问权限的 S3 用户。

流程

1. 创建新数据池：

语法

```
ceph osd pool create POOL_NAME
```

示例

```
[ceph: root@host01 /]# ceph osd pool create test.hot.data
```

2. 添加新存储类：

语法

```
radosgw-admin zonegroup placement add  --rgw-zonegroup default --placement-id
PLACEMENT_TARGET --storage-class STORAGE_CLASS
```

示例

```
[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"
        ]
    }
}
```

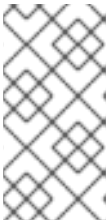
3. *为新存储类提供区放置信息：*

语法

```
radosgw-admin zone placement add --rgw-zone default --placement-id
PLACEMENT_TARGET --storage-class STORAGE_CLASS --data-pool DATA_POOL
```

示例

```
[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
        }
```



注意

在创建具有写入一次的冷或归档数据存储池时，请考虑设置 **compression_type**。

4. 在数据池中启用 **rgw** 应用程序：

语法

```
ceph osd pool application enable POOL_NAME rgw
```

示例

```
[ceph: root@host01 /]# ceph osd pool application enable test.hot.data rgw
enabled application 'rgw' on pool 'test.hot.data'
```

5. 重新启动所有 **rgw** 守护进程。

6. 创建存储桶：

示例

```
[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. **添加对象：**

示例

```
[ceph: root@host01 /]# aws --endpoint=http://10.0.0.80:8080 s3api put-object --bucket
testbucket10  --key compliance-upload --body /root/test2.txt
```

8. **创建第二个数据池：**

语法

```
ceph osd pool create POOL_NAME
```

示例

```
[ceph: root@host01 /]# ceph osd pool create test.cold.data
```

9. **添加新存储类：**

语法

```
radosgw-admin zonegroup placement add  --rgw-zonegroup default --placement-id
PLACEMENT_TARGET --storage-class STORAGE_CLASS
```

示例

```
[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. *为**新存储类提供区放置信息：***

语法

```
radosgw-admin zone placement add --rgw-zone default --placement-id
PLACEMENT_TARGET --storage-class STORAGE_CLASS --data-pool DATA_POOL
```

示例

```
[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

11. *在数据池中启用 rgw 应用程序：*

语法

```
ceph osd pool application enable POOL_NAME rgw
```

示例

```
[ceph: root@host01 /]# ceph osd pool application enable test.cold.data rgw
enabled application 'rgw' on pool 'test.cold.data'
```

12. *重新启动所有 rgw 守护进程。*

13. *要查看 zone group 配置，请运行以下命令：*

语法

```
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. *要查看区配置，请运行以下命令：*

语法

```
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. *创建存储桶：*

示例

```
[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. *在转换前列出对象：*

示例

示例

```
[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. 为生命周期配置创建 **JSON** 文件：

示例

```
[ceph: root@host01 /]# vi lifecycle.json
```

18. 在文件中添加特定的生命周期配置规则：

示例

```
{
    "Rules": [
        {
            "Filter": {
                "Prefix": ""
            },
            "Status": "Enabled",
            "Transitions": [
```

```
            {
                "Days": 5,
                "StorageClass": "hot.test"
            },
 {
                "Days": 20,
                "StorageClass": "cold.test"
            }
        ],
        "Expiration": {
            "Days": 365
        },
        "ID": "double transition and expiration"
    }
  ]
}
```

生命周期配置示例显示一个对象，它将在 5 天后从默认 STANDARD 存储类过渡到 hot.test 存储类，在 20 天后过渡到 cold.test 存储类，并在冷 .test 存储类中最终 365 天后过期。

19. **在存储桶上设置生命周期配置：**

示例

```
[ceph: root@host01 /]# aws s3api put-bucket-lifecycle-configuration --bucket testbucket10 --
lifecycle-configuration file://lifecycle.json
```

20. **检索存储桶上的生命周期配置：**

示例

```
[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. *验证对象是否已转换为给定存储类：*

示例

```
[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"
        }
    }
```

其它资源

- *如需了解有关 bucket 生命周期的详细信息，请参阅 Red Hat Ceph Storage 开发人员指*

南。

8.8.8. 为 S3 启用对象锁定

使用 S3 对象锁定机制，您可以使用保留周期、法律保存和存储桶配置等对象锁定概念来实现 Write-Once-Read-Many (WORM)功能的一部分，作为自定义工作流覆盖数据删除权限的一部分。

重要

对于对象锁定正确执行来支持 GOVERNANCE 或 COMPLIANCE 模式，定义并需要的值是对象版本而不是对象名称。您需要知道对象在写入时的版本，以便可以稍后检索它。

先决条件

- 正在运行的 Red Hat Ceph Storage 集群安装有 Ceph 对象网关。
- Ceph 对象网关节点的根级别访问权限.
- 具有 version-bucket 创建访问权限的 S3 用户。

流程

1. 创建启用了对象锁定的存储桶：

语法

```
aws --endpoint=http://RGW_PORT:8080 s3api create-bucket --bucket BUCKET_NAME --
object-lock-enabled-for-bucket
```

示例

```
[root@rgw-2 ~]# aws --endpoint=http://rgw.ceph.com:8080 s3api create-bucket --bucket
worm-bucket --object-lock-enabled-for-bucket
```

2. 为存储桶设置保留周期：

语法

```
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 }}}'
```

示例

```
[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 }}}'
```

注意

您可以选择 S3 对象锁定中的 RETENTION_MODE 的 GOVERNANCE 或 COMPLIANCE 模式，来对受对象锁定保护的任何对象版本应用不同级别的保护。

在 GOVERNANCE 模式中，用户无法覆盖或删除对象版本，或者更改其锁定设置，除非他们具有特殊权限。

在 COMPLIANCE 模式中，任何用户都无法覆盖或删除受保护的对象版本，包括 AWS 帐户中的 root 用户。当对象被锁定在 COMPLIANCE 模式下时，其 RETENTION_MODE 无法更改，其保留周期无法缩短。COMPLIANCE 模式有助于确保在期间内无法覆盖或删除对象版本。

3. **使用保留时间集将对象放入存储桶中：**

语法

```
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
```

示例

```
[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. **使用同一密钥上传新对象：**

语法

```
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
```

示例

```
[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"
}
```

命令行选项

- **对对象版本设置对象锁定法律：**

示例

```
[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
```

注意

使用对象锁定法律持有操作，您可以将法律持有在对象版本上，从而防止对象版本被覆盖或删除。法律持有没有关联的保留周期，因此请在删除前保持有效。

- **列出存储桶中的对象，以仅检索对象的最新版本：**

示例

```
[root@rgw-2 ~]# aws --endpoint=http://rgw.ceph.com:8080 s3api list-objects --bucket worm-bucket
```

- *列出存储桶中的对象版本：*

示例

```
[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"
            }
            }
        }
    ]
}
```

- *使用 **version-ids** 访问对象：*

示例

```
[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 对象网关日志各个用户的使用情况。您还可以在日期范围内跟踪用户使用情况。

选项包括：

- 起始日期： --start-date 选项允许您从特定开始日期过滤使用情况统计信息(格式: yyyy-mm-dd[HH:MM:SS])。
- 结束日期： --end-date 选项允许您过滤特定日期的使用情况(格式: yyyy-mm-dd[HH:MM:SS])。
- 日志条目： --show-log-entries 选项允许您指定是否使用使用 stats 包括日志条目(选项: true | false)。

注意

您可以使用分钟和秒指定时间，但以 1 小时分辨率存储。

8.9.1. 显示用法

要显示使用情况统计信息，请指定 用量显示。要显示特定用户的使用情况，您必须指定用户 ID。您还可以指定起始日期和结束日期，以及是否显示日志条目。

示例

```
[ceph: root@host01 /]# radosgw-admin usage show \
        --uid=johndoe --start-date=2022-06-01 \
        --end-date=2022-07-01
```

您还可以通过省略用户 ID 来显示所有用户的使用情况信息摘要。

示例

```
[ceph: root@host01 /]# radosgw-admin usage show --show-log-entries=false
```

8.9.2. Trim usage

在大量使用时，使用情况日志可能会占用大量存储空间。您可以为所有用户和特定用户修剪使用日志。您还可以指定 Trim 操作的日期范围。

示例

```
[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 对象网关数据布局

虽然 RADOS 仅了解具有其扩展属性(xattrs)和对象映射(OMAP)的池和对象，但概念上 Ceph 对象网关将其数据组织为三种不同的类型：

- 元数据
- bucket 索引
- data

元数据

元数据有三个部分：

- 用户 ：保存用户信息。
- bucket ：包含存储桶名称和 bucket 实例 ID 之间的映射。
- bucket.instance ：保存存储桶实例信息。

您可以使用以下命令查看元数据条目：

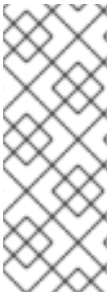
语法

```
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
```

示例

```
[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
```

每个元数据条目都保留在单个 RADOS 对象上。



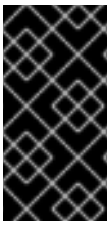
注意

Ceph 对象网关对象可能由多个 RADOS 对象组成，第一个是包含元数据的头，如清单、访问控制列表(ACL)、内容类型、ETag 和用户定义的元数据。元数据存储在 xattrs 中。头还可能包含最多 512 KB 的对象数据，以提高效率和原子性。清单描述各个对象在 RADOS 对象中布局的方式。

bucket 索引

它是不同类型的元数据，并单独保存。bucket 索引在 RADOS 对象中包含键值映射。默认情况下，每个 bucket 是一个 RADOS 对象，但可以在多个 RADOS 对象上对映射进行分片。

映射本身保存在与每个 RADOS 对象关联的 OMAP 中。每个 OMAP 的键是对象的名称，值包含该对象的一些基本元数据，在列出存储桶时显示的元数据。每个 OMAP 均包含一个标头，并在该标头中保留一些存储桶核算元数据，如对象数量、总大小等。



重要

在使用 radosgw-admin 工具时，请确保工具和 Ceph 集群具有相同的版本。不支持使用不匹配的版本。

注意

OMAP 是一个键值存储，与对象关联，类似于与 POSIX 文件关联的扩展属性。对象的 OMAP 不物理位于对象的存储中，但其精确实施对 Ceph 对象网关不可见且不可更改。

data

对象数据保存在每个 Ceph 对象网关对象的一个或多个 RADOS 对象中。

8.10.1. 对象查找路径

在访问对象时，REST API 附带三个参数的 Ceph 对象网关：

- 帐户信息，在 Swift 中有 S3 或帐户名称中的访问密钥
- 存储桶或容器名称

- *对象名称或密钥*

目前，Ceph 对象网关仅使用帐户信息来查找用户 ID 和访问控制。它仅使用 bucket 名称和对象键来寻址池中的对象。

帐户信息

Ceph 对象网关中的用户 ID 是一个字符串，通常是来自用户凭证的实际用户名，而不是散列或映射的标识符。

在访问用户数据时，用户记录将从带有 users.uid 命名空间的 default.rgw.meta 池中的对象 USER_ID 加载。.Bucket 名称 They 在带有 root 命名空间的 default.rgw.meta 池中表示。加载存储桶记录，以获取一个标记，它充当存储桶 ID。

对象名称

对象位于 default.rgw.buckets.data 池中。对象名称是 MARKER_KEY，如 default.7593.4_image.png，其中标记为 default.7593.4，键是 image.png。这些串联的名称不会被解析，并且仅传递到 RADOS。因此，选择分隔符并不重要，不会造成不确定。因此，对象名称中允许斜杠，如键。

8.10.1.1. 多个数据池

可以创建多个数据池，使得不同用户的 bucket 默认在不同的 RADOS 池中创建，从而提供必要的扩展。这些池的布局和命名由策略设置控制。

8.10.2. 存储桶和对象列表

属于给定用户的 bucket 在名为 USER_ID.buckets 的对象的 OMAP 中列出，例如 foo.buckets，位于带有 users.uid 命名空间的 default.rgw.meta 池中。在列出存储桶时，在更新存储桶时访问这些对象，并更新和检索存储桶统计信息，如配额。这些列表与 .rgw 池中的 bucket 保持一致。

注意

有关这些 OMAP 条目的值，请参阅用户可见的、编码的类 cls_user _bucket 及其嵌套类 cls_user_bucket。

属于给定存储桶的对象列在存储桶索引中。索引对象的默认命名在 default.rgw.buckets.index 池中是

.dir.MARKER。

其它资源

- 如需了解更多详细信息，请参阅 Red Hat Ceph Storage 对象网关指南中的 配置存储桶索引重新划分 部分。

8.10.3. 对象网关数据布局参数

这是 Ceph 对象网关的数据布局参数列表。

已知的池：

.rgw.root

每个对象未指定的区域、区域和全局信息记录。

ZONE.rgw.control

notify.N

ZONE.rgw.meta

具有不同元数据的多个命名空间

Namespace: root

BUCKET .bucket.meta.BUCKET:MARKER # see put_bucket_instance_info ()

租户用于忽略存储桶，但不用于 bucket 实例。

示例

```
.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

在 USER 对象中包括每个用户信息 (RGWUserInfo)，以及 USER.buckets 对象的 omaps 中的每个用户的存储桶列表。如果非空，USER 可能会包含租户。

示例

```
prodtx$prodt
test2.buckets
prodtx$prodt.buckets
test2
```

namespace: users.email

Unimportant

namespace: users.keys

47UA98JSTJZ9YAN3OS3O

这允许 Ceph 对象网关在身份验证过程中按其访问密钥查找用户。

namespace: users.swift

test:tester

ZONE.rgw.buckets.index

对象名为 .dir.MARKER，每个对象都包含存储桶索引。如果索引被分片，每个分片会在标记后附加分片索引。

ZONE.rgw.buckets.data

default.7593.4__shadow_.488urDFerTYXavx4yAd-Op8mxehnvTI_1 MARKER_KEY

一个标记示例为 default.16004.1 或 default.7593.4。当前格式为

ZONE.INSTANCE_ID.BUCKET_ID，但生成后，不会再次解析标记，因此其格式可能会在以后自由更改。

其它资源

- 如需了解更多详细信息，请参阅 Red Hat Ceph Storage 对象网关指南中的 Ceph 对象网关数据布局 。

8.11. 最接近数据的速率限值

作为存储管理员，您可以在使用 Ceph 对象网关配置在 Red Hat Ceph Storage 集群中保存对象时，根据操作和带宽设置用户和存储桶的速率限制。

8.11.1. 存储集群中的速率限制的目的

您可以在 Ceph 对象网关配置中设置用户和存储桶的速率限制。速率限制包括最大读取操作数、每分钟的写入操作数，以及每分钟可写入或读取的字节数或为每个存储桶读取。

在 REST 中使用 GET 或 HEAD 方法的请求是"读取请求"，否则它们是"写入请求"。

Ceph 对象网关分别跟踪用户和 bucket 请求，不与其他网关共享，这意味着所需的配置的限制应除以活动对象网关的数量。

例如，如果用户 A 应每分钟限制为十个 ops，并且集群中有两个 Ceph 对象网关，则用户 A 的限制应该为五，即对于两个 Ceph 对象网关每分钟为十个 opss。如果 Ceph 对象网关之间没有平衡请求，则速率限制可能不足。例如，如果 ops 的限制是五，并且有两个 Ceph 对象网关，但负载平衡器仅向其中一个 Ceph 对象网关发送负载，则有效限制将是五个 ops，因为这个限制针对一个 Ceph 对象网关实施。

如果存储桶达到了限制，但没有供用户使用，反之亦然请求也会被取消。

请求被接受后发生带宽数。因此，即使存储桶或用户在请求期间达到其带宽限制，此请求也会进行。

Ceph 对象网关保留超过配置的值的"debt"，并防止此用户或 bucket 发送更多请求，直到其"debt"被支付。"debt"最大大小是每分钟的 max-read/write-bytes 两倍。如果用户 A 每分钟有 1 字节读取限制，且此用户会尝试 GET 1 GB 对象，用户可以执行此操作。

用户 A 完成此 1 GB 操作后，Ceph 对象网关将阻止用户请求最多两分钟，直到用户 A 能够再次发送 GET 请求。

限制率的不同选项：

- bucket： --bucket 选项允许您为存储桶指定速率限制。
- user: --uid 选项允许您为用户指定速率限制。
- 最大读取 ops： --max-read-ops 设置允许您指定每个 Ceph 对象网关每分钟读取 ops 的最大数量。0 代表禁用此设置，即无限访问。
- 最大读取字节数： --max-read-bytes 设置允许您指定每个 Ceph 对象网关每分钟的最大读取字节数。0 代表禁用此设置，即无限访问。
- 最大写入 ops ： --max-write-ops 设置允许您指定每个 Ceph 对象网关每分钟的最大写入 ops 数。0 代表禁用此设置，即无限访问。
- 最大写入字节数： --max-write-bytes 设置允许您指定每个 Ceph 对象网关每分钟的最大写入字节数。0 代表禁用此设置，即无限访问。
- 速率限制范围： --rate-limit-scope 选项设置速率限制的范围。选项包括 bucket、用户 和 匿名。bucket 速率限制应用到存储桶，用户速率限制应用到用户，匿名则应用到未经身份验证的用户。匿名范围仅适用于全局速率限制。

8.11.2. 启用用户速率限制

您可以在 Ceph 对象网关配置中设置用户的速率限制。用户的限制包括最大读取操作数、每分钟写入操作数，以及每分钟可写入或读取的字节数。

当使用带有将 ratelimit-scope 设置为 user 的 radosgw-admin ratelimit set 命令，在设置速率限制值后您可以对用户启用速率限制。

先决条件

- 一个正在运行的 Red Hat Ceph Storage 集群。
- 安装了 Ceph 对象网关。

流程

1. 为用户设置速率限制：

语法

```
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]
```

示例

```
[ceph: root@host01 /]# radosgw-admin ratelimit set --ratelimit-scope=user --uid=testing --max-read-ops=1024 --max-write-bytes=10240
```

如果 NUMBER_OF_OPERATIONS 或 NUMBER_OF_BYTES 的值为 0，代表禁用特定的速率限制属性检查。

2. 获取用户速率限制：

语法

```
radosgw-admin ratelimit get --ratelimit-scope=user --uid=USER_ID
```

示例

```
[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. 启用用户速率限制：

语法

```
radosgw-admin ratelimit enable --ratelimit-scope=user --uid=USER_ID
```

示例

```
[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. 可选：禁用用户速率限制：

语法

```
radosgw-admin ratelimit disable --ratelimit-scope=user --uid=USER_ID
```

示例

```
[ceph: root@host01 /]# radosgw-admin ratelimit disable --ratelimit-scope=user --uid=testing
```

8.11.3. 启用存储桶速率限制

您可以在 Ceph 对象网关配置中设置存储桶的速率限制。bucket 的速率限制包括最大读取操作数、每分钟写入操作数，以及每分钟可写入或读取的字节数。

在设置速率限制值后，您可以使用 radosgw-admin ratelimit set 命令并将 ratelimit-scope 设置为 bucket 后，对存储桶启用速率限制。

先决条件

- 一个正在运行的 Red Hat Ceph Storage 集群。
- 安装了 Ceph 对象网关。

流程

1.

1. *为存储桶设置速率限制：*

语法

```
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]
```

示例

```
[ceph: root@host01 /]# radosgw-admin ratelimit set --ratelimit-scope=bucket --bucket=mybucket --max-read-ops=1024 --max-write-bytes=10240
```

如果 NUMBER_OF_OPERATIONS 或 NUMBER_OF_BYTES 的值为 0，代表禁用特定的速率限制属性检查。

2. *获取存储桶速率限制：*

语法

```
radosgw-admin ratelimit get --ratelimit-scope=bucket --bucket=BUCKET_NAME
```

示例

```
[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. ***启用存储桶速率限制：***

语法

```
radosgw-admin ratelimit enable --ratelimit-scope=bucket --bucket=BUCKET_NAME
```

示例

```
[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. ***可选：禁用存储桶速率限制：***

语法

```
radosgw-admin ratelimit disable --ratelimit-scope=bucket --bucket=BUCKET_NAME
```

示例

```
[ceph: root@host01 /]# radosgw-admin ratelimit disable --ratelimit-scope=bucket --
bucket=mybucket
```

8.11.4. 配置全局速率限制

您可以在期间配置内读取或写入全局速率限制设置。您可以通过带有 **global ratelimit** 参数的全局速率限制设置来覆盖用户或存储桶的速率限制配置，这与 **ratelimit set, ratelimit enable,** 和 **ratelimit disable** 命令相对应。

注意

在存在 **realm** 和 **period** 的多站点配置中，必须使用 **period update --commit** 命令提交对全局速率限制的更改。如果没有 **period**，则必须重启 **Ceph** 对象网关，才能使更改生效。

先决条件

- 一个正在运行的 **Red Hat Ceph Storage** 集群。
- 安装了 **Ceph** 对象网关。

流程

1. 查看全局速率限制设置：

语法

```
radosgw-admin global ratelimit get
```

示例

```
[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. **为存储桶配置和启用速率限制范围：**

   a. **为存储桶设置全局速率限制：**

      语法

```
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]
```

示例

```
[ceph: root@host01 /]# radosgw-admin global ratelimit set --ratelimit-scope bucket --max-read-ops=1024
```

b. 启用存储桶速率限制：

语法

```
radosgw-admin global ratelimit enable --ratelimit-scope=bucket
```

示例

```
[ceph: root@host01 /]# radosgw-admin global ratelimit enable --ratelimit-scope bucket
```

3. 为经过身份验证的用户配置和启用速率限制范围：

a. *为用户设置全局速率限制：*

语法

```
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]
```

示例

```
[ceph: root@host01 /]# radosgw-admin global ratelimit set --ratelimit-scope=user --max-read-ops=1024
```

b. *启用用户速率限制：*

语法

```
radosgw-admin global ratelimit enable --ratelimit-scope=user
```

示例

```
[ceph: root@host01 /]# radosgw-admin global ratelimit enable --ratelimit-scope=user
```

4. *为未经身份验证的用户配置和启用速率限制范围：*

   a. *为未经身份验证的用户设置全局速率限制：*

   语法

   ```
   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]
   ```

   示例

   ```
   [ceph: root@host01 /]# radosgw-admin global ratelimit set --ratelimit-scope=anonymous --max-read-ops=1024
   ```

   b. *启用用户速率限制：*

   语法

   ```
   radosgw-admin global ratelimit enable --ratelimit-scope=anonymous
   ```

   示例

   ```
   [ceph: root@host01 /]# radosgw-admin global ratelimit enable --ratelimit-scope=anonymous
   ```

8.12. 优化 CEPH 对象网关的垃圾回收

当新数据对象写入到存储集群中时，Ceph 对象网关会立即为这些新对象分配存储。在删除或覆盖存储集群中的数据对象后，Ceph 对象网关将从存储桶索引中删除这些对象。之后一段时间后，Ceph 对象网关会清除用于存储存储群集中对象的空间。从存储集群中清除已删除对象数据的过程称为 Garbage Collection 或 GC。

垃圾回收操作通常在后台运行。您可以将这些操作配置为持续运行，或者仅在低活动和轻量级工作负载期间运行。默认情况下，Ceph 对象网关持续执行 GC 操作。由于 GC 操作是 Ceph 对象网关操作的一个正常部分，因此大部分时间都存在符合垃圾回收条件的已删除对象。

8.12.1. 查看垃圾回收队列

在从存储集群中清除被删除和覆盖对象前，useradosgw-admin 来查看等待垃圾回收的对象。

先决条件

- 一个正在运行的 Red Hat Ceph Storage 集群。
- Ceph 对象网关的根级别访问权限.

流程

- 查看等待垃圾回收的对象队列：

示例

```
[ceph: root@host01 /]# radosgw-admin gc list
```

注意

要列出队列中的所有条目，包括未过期条目，请使用 --include-all 选项。

8.12.2. 调整 Garbage Collection 设置

Ceph 对象网关为新的和覆盖的对象立即分配存储。此外，多部分上传的部分也消耗了一些存储。

从 bucket 索引中删除对象后，Ceph 对象网关会清除用于已删除对象的存储空间。类似地，Ceph 对象网关将在多部分上传完成后删除与多部分上传相关的数据，或者上传未激活或未能完成时可配置的时间。从 Red Hat Ceph Storage 集群清除已删除对象数据的过程称为垃圾回收(GC)。

可使用以下命令查看等待垃圾回收的对象：

```
radosgw-admin gc list
```

垃圾回收是一种后台活动，它持续或低负载期间运行，具体取决于存储管理员配置 Ceph 对象网关的方式。默认情况下，Ceph 对象网关持续执行垃圾收集操作。由于垃圾回收操作是 Ceph 对象网关的常规功能，特别是对象删除操作，因此大多数时间都存在符合垃圾回收条件的对象。

某些工作负载可临时或永久超过垃圾收集活动的速度。对于删除密集型工作负载而言，这尤其适用，其中很多对象都会在短时间内存储，然后将其删除。对于这些类型的工作负载，存储管理员可以使用以下配置参数，相对于其他操作增加垃圾回收操作的优先级：

- rgw_gc_obj_min_wait 配置选项在清除已删除对象数据前等待最短时间（以秒为单位）。默认值为 2 小时或 7200 秒。对象不会立即清除，因为客户端可能会读取对象。在高负载中，此设置可能会消耗太多存储，或者有大量已删除的对象可以清除。红帽建议不要在 30 分钟内设置这个值，或设置 1800 秒。

- rgw_gc_processor_period 配置选项是垃圾回收周期的运行时间。也就是说，连续运行垃圾回收线程之间的时间长度。如果垃圾回收的时间超过这一时间段，Ceph 对象网关不会在再次运行垃圾回收循环前等待。

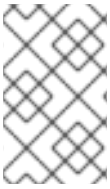
- rgw_gc_max_concurrent_io 配置选项指定网关垃圾回收线程在清除已删除数据时使用的最大并发 IO 操作数。在删除重度工作负载时，请考虑将此设置增加到更多并发 IO 操作。

- rgw_gc_max_trim_chunk 配置选项指定要在单个操作中从垃圾收集器日志中移除的最大密

钥数。在删除重量操作时，请考虑增加密钥的最大数量，以便在每次垃圾收集操作期间清除更多对象。

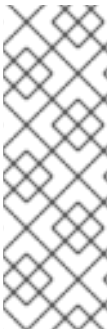
从 Red Hat Ceph Storage 4.1 开始，从垃圾回收日志中卸载索引对象的 OMAP 有助于降低垃圾回收活动对存储集群的性能影响。在 Ceph 对象网关中添加了一些新配置参数来调优垃圾回收队列，如下所示：

- rgw_gc_max_deferred_entries_size 配置选项设置垃圾回收队列中延迟条目的最大大小。
- rgw_gc_max_queue_size 配置选项设置用于垃圾回收的最大队列大小。这个值不应大于 osd_max_object_size 减 rgw_gc_max_deferred_entries_size 减 1 KB。
- rgw_gc_max_deferred 配置选项设置垃圾回收队列中存储的最大延迟条目数。



注意

这些垃圾回收配置参数适用于 Red Hat Ceph Storage 7 及更高版本。



注意

在测试中，存储群集使用均匀均衡的删除写入工作负载（如 50% 删除和 50% 的写入操作），该存储群集占用了 11 小时。这是因为 Ceph 对象网关垃圾回收无法与删除操作同步。如果发生这种情况，集群状态将切换到 HEALTH_ERR 状态。对并行垃圾回收可调项的主动设置会显著延迟存储群集的测试，可能对很多工作负载有用。典型的实际存储集群工作负载可能无法导致存储集群主要因为垃圾回收而填满。

8.12.3. 为删除密集型工作负载调整垃圾回收

有些工作负载可能会临时或永久超过垃圾收集活动的速度。对于删除密集型工作负载而言，这尤其适用，其中很多对象都会在短时间内存储，然后将其删除。对于这些类型的工作负载，请考虑增加垃圾回收操作相对于其他操作的优先级。如有关于 Ceph 对象网关粘合的其他问题，请联系红帽支持团队。

先决条件

- 一个正在运行的 Red Hat Ceph Storage 集群。

- 对存储集群中所有节点的根级别访问权限。

流程

1. 将 rgw_gc_max_concurrent_io 的值设置为 20，将 rgw_gc_max_trim_chunk 的值设置为 64：

   示例

```
[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. 重新启动 Ceph 对象网关，以允许更改的设置生效。

3. 在 GC 活动期间监控存储集群，以验证增加的值不会影响性能。

重要

切勿修改正在运行的集群中的 rgw_gc_max_objs 选项的值。您应该仅在部署 RGW 节点前更改此值。

其它资源

- Ceph RGW - GC Tuning Options
- RGW 常规设置
- 配置参考

8.13. 优化 CEPH 对象网关的数据对象存储

bucket 生命周期配置优化了数据对象存储，以提高其效率，并在数据生命周期内提供有效的存储。

Ceph 对象网关中的 S3 API 目前支持 AWS 存储桶生命周期配置操作的子集：

- 过期
- NoncurrentVersionExpiration
- AbortIncompleteMultipartUpload

先决条件

- 一个正在运行的 Red Hat Ceph Storage 集群。
- 对存储集群中所有节点的根级别访问权限。

8.13.1. bucket 生命周期并行线程处理

Ceph 对象网关现在允许在多个 Ceph 对象网关实例之间并行处理 bucket 生命周期。增加并行运行的线程数量，使 Ceph 对象网关能够更有效地处理大量工作负载。此外，Ceph 对象网关现在使用数字序列进行索引分片枚举，而不使用顺序编号。

8.13.2. 优化存储桶生命周期

Ceph 配置文件中的两个选项会影响存储桶生命周期处理效率：

- rgw_lc_max_worker 指定并行运行的生命周期 worker 线程数量。这可同时处理存储桶和索引分片。这个选项的默认值为 3。
- rgw_lc_max_wp_worker 指定每个生命周期 worker 线程池中的线程数量。此选项有助于加快每个存储桶的处理。这个选项的默认值为 3。

例如，对于 bucket 数量较多的工作负载， Thoterment 增加了 rgw_lc_max_worker 选项的值。

对于 bucket 数量较少但每个 bucket 中数量较多的工作负载，比如在成百上百上千 >-Asconsider 中增加了 rgw_lc_max_wp_worker 选项的值。

注意

在增加其中一个选项的值之前，请验证当前存储集群性能和 Ceph 对象网关的利用率。红帽不建议为其中一个选项分配 10 或以上值。

先决条件

- 一个正在运行的 Red Hat Ceph Storage 集群。
- 对存储集群中所有节点的根级别访问权限。

流程

1. 要增加并行运行的线程数量，请将 rgw_lc_max_worker 的值设置为 3 到 9 之间的值：

示例

```
[ceph: root@host01 /]# ceph config set client.rgw rgw_lc_max_worker 7
```

2. 要增加每个线程工作池中的线程数量，请将 rgw_lc_max_wp_worker 的值设置为 3 到 9 之间的值：

示例

```
[ceph: root@host01 /]# ceph config set client.rgw rgw_lc_max_wp_worker 7
```

3. 重新启动 Ceph 对象网关，以允许更改的设置生效。

4. 监控存储集群，以验证增加的值不会影响性能。

其它资源

- 有关存储桶生命周期和并行线程处理的更多信息，请参阅 Bucket 生命周期并行处理
- 有关 Ceph 对象网关生命周期的更多信息，请联系红帽支持。

8.14. 将数据转换到 AMAZON S3 云服务

您可以将数据迁移到远程云服务，作为使用存储类的生命周期配置的一部分，以降低成本并改进易管理性。转换是单向的，无法从远程区转换数据。此功能是启用数据转换到多个云供应商，如 Amazon (S3)。

使用 cloud-s3 作为层类型，配置需要将数据转换到的远程云 S3 对象存储服务。它们不需要数据池，并在 zonegroup 放置目标中定义。

先决条件

- 安装了 Ceph 对象网关的 Red Hat Ceph Storage 集群。
- 远程云服务 Amazon S3 的用户凭据。
- 在 Amazon S3 上创建的目标路径。
- 在 bootstrapped 节点上安装 s3cmd。
- Amazon AWS 在本地配置为下载数据。

流程

1. **创建带有访问密钥和 secret 密钥的用户：**

语法

```
radosgw-admin user create --uid=USER_NAME --display-name="DISPLAY_NAME" [--access-key ACCESS_KEY --secret-key SECRET_KEY]
```

示例

```
[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. **在 bootstrapped 节点上，添加层类型为 cloud-s3 的存储类：**

注意

使用 --tier-type=cloud-s3 选项创建存储类后，无法稍后将其修改为任何其他存储类类型。

语法

```
radosgw-admin zonegroup placement add --rgw-zonegroup =ZONE_GROUP_NAME \
                      --placement-id=PLACEMENT_ID \
                      --storage-class =STORAGE_CLASS_NAME \
                      --tier-type=cloud-s3
```

示例

```
[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. *更新 **storage_class** ：*

注意

*如果集群是多站点设置的一部分，请运行 **period update --commit**，以便 **zonegroup** 更改传播到多站点中的所有区域。*

注意

*确保 **access_key** 和 **secret** 不以数字开头。*

必需的参数为：

- ***access_key** 是用于特定连接的远程云 **S3** 访问密钥。*

- *secret 是远程云 S3 服务的 secret 密钥。*
- *endpoint 是远程云 S3 服务端点的 URL。*
- *区域 （用于 AWS）是远程云 S3 服务区域名称。*

可选参数：

- *target_path 定义目标路径的创建方式。目标路径指定源 bucket-name/object-name 附加到的前缀。如果没有指定，则创建的 target_path 是 rgwx-ZONE_GROUP_NAME-STORAGE_CLASS_NAME-cloud-bucket。*
- *target_storage_class 定义了一个对象传输到的目标存储类。如果没有指定，则对象将转换为 STANDARD 存储类。*
- *retain_head_object （如果为 true）保留对象转换为云的元数据。如果为 false （默认），对象会在转换后删除。对于当前版本的对象，会忽略这个选项。*
- *multipart_sync_threshold 指定此大小或更大对象使用多部分上传转换为云。*
- *multipart_min_part_size 指定在使用多部分上传转换对象时要使用的最小部分大小。*

语法

```
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
```

示例

```
[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. ***重启 Ceph 对象网关：***

语法

```
ceph orch restart CEPH_OBJECT_GATEWAY_SERVICE_NAME
```

示例

```
[ceph: root@host 01 /]# ceph orch restart rgw.rgw.1

Scheduled to restart rgw.rgw.1.host03.vkfldf on host 'host03'
```

5. ***退出 shell 和以 root 用户身份，在 bootstrapped 节点上配置 Amazon S3：***

示例

```
[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. ***创建 S3 存储桶：***

语法

```
s3cmd mb s3://NAME_OF_THE_BUCKET_FOR_S3
```

示例

```
[root@host01 ~]# s3cmd mb s3://awstestbucket
Bucket 's3://awstestbucket/' created
```

7. **创建文件，输入所有数据，并将其移动到 S3 服务：**

语法

```
s3cmd put FILE_NAME  s3://NAME_OF_THE_BUCKET_ON_S3
```

示例

```
[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. **创建生命周期配置转换策略：**

语法

```
<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>
```

示例

```
[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. **设置生命周期配置转换策略：**

语法

```
s3cmd setlifecycle FILE_NAME s3://NAME_OF_THE_BUCKET_FOR_S3
```

示例

```
[root@host01 ~]#  s3cmd setlifecycle lc_config.xml s3://awstestbucket

s3://awstestbucket/: Lifecycle Policy updated
```

10. ***登录 cephadm shell ：***

示例

```
[root@host 01 ~]# cephadm shell
```

11. ***重启 Ceph 对象网关：***

语法

```
ceph orch restart CEPH_OBJECT_GATEWAY_SERVICE_NAME
```

示例

```
[ceph: root@host 01 /]# ceph orch restart rgw.rgw.1

Scheduled to restart rgw.rgw.1.host03.vkfldf on host 'host03'
```

验证

1. ***在源集群中，使用 radosgw-admin lc list 命令验证数据是否已移到 S3：***

   示例

```
[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. ***验证对象在云端点上的转换：***

   示例

```
[root@client ~]$ radosgw-admin bucket list
[
    "awstestbucket"
]
```

3. ***列出存储桶中的对象：***

   示例

```
[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. ***列出 S3 存储桶的内容：***

示例

```
[root@host01 ~]# s3cmd ls s3://awstestbucket
2022-08-25 09:57            0  s3://awstestbucket/test.txt
```

5. ***检查该文件的信息：***

示例

```
[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. ***从 Amazon S3 本地下载数据：***

   a. ***配置 AWS：***

   示例

   ```
   [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. ***列出 AWS 存储桶的内容：***

   示例

   ```
   [client@client01 ~]$ aws s3 ls s3://dfqe-bucket-01/awstest
   PRE awstestbucket/
   ```

   c. ***从 S3 下载数据：***

   示例

```
[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. 将数据转换为 AZURE 云服务

您可以将数据迁移到远程云服务，作为使用存储类的生命周期配置的一部分，以降低成本并改进易管理性。转换是单向的，无法从远程区转换数据。此功能是启用数据转换到多个云供应商，如 Azure。与 AWS 配置相关的一个关键区别是您需要配置多云网关(MCG)，并使用 MCG 从 S3 协议转换为 Azure Blob。

使用 cloud-s3 作为层类型，配置需要将数据转换到的远程云 S3 对象存储服务。它们不需要数据池，并在 zonegroup 放置目标中定义。

先决条件

- 安装了 Ceph 对象网关的 Red Hat Ceph Storage 集群。
- 远程云服务 Azure 的用户凭证。
- Azure 配置在本地来下载数据。
- 在 bootstrapped 节点上安装 s3cmd。
- 为 MCG 命名空间创建的 Azure 容器。在本例中，它是 mcgnamespace。

流程

1. 创建带有访问密钥和 secret 密钥的用户：

语法

```
radosgw-admin user create --uid=USER_NAME --display-name="DISPLAY_NAME" [--access-key ACCESS_KEY --secret-key SECRET_KEY]
```

示例

```
[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. *作为 **root** 用户，使用用户凭证配置 **AWS CLI**，并使用默认放置创建存储桶：*

语法

```
aws s3 --ca-bundle CA_PERMISSION --profile rgw  --endpoint ENDPOINT_URL --region
default mb s3://BUCKET_NAME
```

示例

```
[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. *验证存储桶是否在放置规则中使用 **default-placement** ：*

示例

```
[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. *使用部署的 **OpenShift Data Foundation (ODF)**登录到 **OpenShift Container Platform***

(OCP)集群：

示例

```
[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. *配置在 Azure 中的 OCP 集群上运行的多云网关(MCG)命名空间 Azure 存储桶：*

语法

```
noobaa namespacestore create azure-blob az --account-key='ACCOUNT_KEY' --account-
name='ACCOUNT_NAME' --target-blob-container='_AZURE_CONTAINER_NAME'
```

示例

```
[root@host01 ~]$ noobaa namespacestore create azure-blob  az --account-
key='iq3+6hRtt9bQ46QfHKQ0nSm2aP+tyMzdn8dBSRW4XWrFhY+1nwfqEj4hk2q66nmD85E/o
5OrrUqo+AStkKwm9w==' --account-name='transitionrgw' --target-blob-
container='mcgnamespace'
```

6. *创建一个 MCG bucket 类，指向 命名空间存储 ：*

示例

```
[root@host01 ~]$ noobaa bucketclass create namespace-bucketclass single aznamespace-bucket-class --resource az -n openshift-storage
```

7. **创建用于过渡到云的对象存储桶声明(OBC)：**

语法

```
noobaa obc create OBC_NAME --bucketclass aznamespace-bucket-class -n openshift-storage
```

示例

```
[root@host01 ~]$ noobaa obc create rgwobc --bucketclass aznamespace-bucket-class -n openshift-storage
```

注意

使用 OBC 提供的凭据，在 Ceph 对象网关上配置 zonegroup 放置。

8. **在 bootstrapped 节点上，在之前在 Azure 中配置了 MCG 的默认放置的默认放置上，在默认放置上创建一个类型为 cloud-s3 的存储类：**

注意

使用 --tier-type=cloud-s3 选项创建存储类后，无法稍后将其修改为任何其他存储类类型。

语法

```
radosgw-admin zonegroup placement add --rgw-zonegroup =ZONE_GROUP_NAME \
                    --placement-id=PLACEMENT_ID \
                    --storage-class =STORAGE_CLASS_NAME \
                    --tier-type=cloud-s3
```

示例

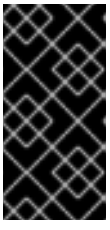
```
[ceph: root@host01 /]# radosgw-admin zonegroup placement add --rgw-zonegroup=default \
                                    --placement-id=default-placement \
                                    --storage-class=AZURE \
                                    --tier-type=cloud-s3
[
    {
        "key": "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. ***配置云 S3 云存储类：***

语法

```
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

将 retain_head_object 参数设置为 true 可保留元数据或对象头，以列出正在转换的对象。

示例

```
[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. ***重启 Ceph 对象网关：***

语法

```
ceph orch restart CEPH_OBJECT_GATEWAY_SERVICE_NAME
```

示例

```
[ceph: root@host 01 /]# ceph orch restart client.rgw.objectgwhttps.host02.udyllp

Scheduled to restart client.rgw.objectgwhttps.host02.udyllp on host 'host02
```

11. 为之前创建的存储桶创建生命周期配置转换策略。在本例中，存储桶正在 转换 ：

语法

```
cat transition.json
{
    "Rules": [
        {
            "Filter": {
                "Prefix": ""
            },
            "Status": "Enabled",
            "Transitions": [
                {
                    "Days": 30,
                    "StorageClass": "STORAGE_CLASS"
                }
            ],
            "ID": "TRANSITION_ID"
        }
    ]
        }
```

注意

bucket 中超过 30 天的所有对象都传送到名为 AZURE 的云存储类。

示例

```
[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. **使用 AWS CLI 应用存储桶生命周期配置：**

语法

```
 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
```

示例

```
[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. ***可选：获取生命周期配置：***

语法

```
 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
```

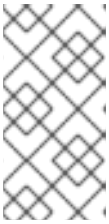
示例

```
[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. ***可选：使用 radosgw-admin lc list 命令获取生命周期配置：***

示例

```
[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"
    }
]
```



注意

UNINITAL 状态表示生命周期配置没有被处理。它在转换过程完成后变为 COMPLETED 状态。

15. **登录 cephadm shell ：**

示例

```
[root@host 01 ~]# cephadm shell
```

16. **重启 Ceph 对象网关守护进程：**

语法

```
ceph orch daemon CEPH_OBJECT_GATEWAY_DAEMON_NAME
```

示例

```
[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. **将数据从源集群迁移到 Azure:**

示例

```
[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. **验证数据的转换：**

示例

```
[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. **使用 rados ls 命令验证数据是否已移到 Azure：**

示例

```
[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. **如果没有转换数据，您可以运行 lc process 命令：**

示例

```
[root@host 01 ~]# radosgw-admin lc process
```

这将强制生命周期过程启动和评估配置的所有存储桶生命周期策略。然后，它会根据需要开始转换数据。

验证

1. **运行 radosgw-admin lc list 命令以验证转换的完成：**

示例

```
[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. **列出存储桶中的对象：**

示例

```
[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. **列出集群中的对象：**

示例

```
[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
```

*对象的大小为 **0**。您可以列出对象，但无法复制它们，因为它们被转换为 **Azure**。*

4. *使用 **S3 API** 检查对象头：*

示例

```
[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"
}
```

*您可以看到存储类已从 **STANDARD** 更改为 **CLOUDTIER**。*

第 9 章 测试

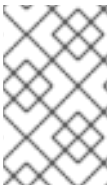
作为存储管理员，您可以执行基本功能测试，以验证 Ceph 对象网关环境是否按预期工作。您可以通过为 S3 接口创建初始 Ceph 对象网关用户，然后为 Swift 接口创建子用户，从而使用 REST 接口。

先决条件

- 一个正常运行的 Red Hat Ceph Storage 集群。
- 安装 Ceph 对象网关软件.

9.1. 创建 S3 用户

若要测试网关，请创建 S3 用户并授予用户访问权限。man radosgw-admin 命令提供关于其他命令选项的信息。



注意

在多站点部署中，始终在 master zone group 的 master zone 中的主机上创建用户。

先决条件

- root 或 sudo 访问权限
- 已安装 Ceph 对象网关

流程

1. 创建 S3 用户：

语法

```
radosgw-admin user create --uid=name --display-name="USER_NAME"
```

使用 S3 用户的名称替换 name ：

示例

```
[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. 验证输出，以确保 access_key 和 secret_key 的值不包含 JSON 转义字符(\)。访问验证需要这些值，但如果值包含 JSON 转义字符，则某些客户端无法处理。要解决这个问题，请执行以下操作之一：

   - 删除 JSON 转义字符。
   - 将字符串封装在引号内。
   - 重新生成密钥，并确保它不包含 JSON 转义字符。
   - 手动指定密钥和 secret。

   不要删除正斜杠 /，因为它是一个有效的字符。

9.2. 创建 SWIFT 用户

要测试 Swift 接口，请创建一个 Swift 子用户。创建 Swift 用户包含两个步骤。第一步是创建用户。第二步是创建机密密钥。

注意

在多站点部署中，始终在 master zone group 的 master zone 中的主机上创建用户。

先决条件

- 安装 Ceph 对象网关.
- Ceph 对象网关节点的根级别访问权限.

流程

1. 创建 Swift 用户：

语法

```
radosgw-admin subuser create --uid=NAME --subuser=NAME:swift --access=full
```

使用 Swift 用户名替换 NAME，例如：

示例

```
[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. *创建 **secret** 密钥：*

语法

```
radosgw-admin key create --subuser=NAME:swift --key-type=swift --gen-secret
```

*使用 **Swift** 用户名替换 **NAME**，例如：*

示例

```
[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. 测试 S3 访问

您需要编写并运行 Python 测试脚本来验证 S3 访问权限。S3 访问测试脚本将连接到 radosgw，创建一个新 bucket 并列出所有存储桶。aws_access_key_id 和 aws_secret_access_key 的值取自 radosgw_admin 命令返回的 access_key 和 secret_key 的值。

先决条件

- *一个正在运行的 Red Hat Ceph Storage 集群。*

- 对节点的根级别访问权限。

流程

1. 为 Red Hat Enterprise Linux 9 启用高可用性软件仓库：

```
subscription-manager repos --enable=rhel-9-for-x86_64-highavailability-rpms
```

2. 安装 python3-boto3 软件包：

```
dnf install python3-boto3
```

3. 创建 Python 脚本：

```
vi s3test.py
```

4. 在文件中添加以下内容：

语法

```
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. 使用配置了网关服务的主机的 URL 替换 endpoint。也就是说，网关主机。确保 host 使用 DNS 解析。使用网关的端口号替换 PORT。

b. 使用 Red Hat Ceph Storage 对象网关指南中的 创建 S3 用户 部分中的 access_key 和 secret_key 值替换 ACCESS 和 SECRET。

5. 运行脚本：

```
python3 s3test.py
```

输出结果类似如下：

```
my-new-bucket 2022-05-31T17:09:10.000Z
```

9.4. 测试 SWIFT 访问

可以通过 swift 命令行客户端验证 Swift 访问权限。命令 man swift 将提供有关可用命令行选项的更多信息。

要安装 swift 客户端，请运行以下命令：

```
sudo yum install python-setuptools
sudo easy_install pip
sudo pip install --upgrade setuptools
sudo pip install --upgrade python-swiftclient
```

要测试 swift 访问，请运行以下命令：

语法

```
# swift -A http://IP_ADDRESS:PORT/auth/1.0 -U testuser:swift -K 'SWIFT_SECRET_KEY' list
```

将 IP_ADDRESS 替换为网关服务器的公共 IP 地址，将 SWIFT_SECRET_KEY 替换为为 swift 用户发布的 radosgw-admin key create 命令的输出中的值。将 PORT 替换为您要使用 Beast 的端口号。如果不替换端口，它将默认为端口 80。

例如：

```
swift -A http://10.10.143.116:80/auth/1.0 -U testuser:swift -K
'244+fz2gSqoHwR3lYtSbIyomyPHf3i7rgSJrF/IA' list
```

输出应该是：

```
my-new-bucket
```

附录 A. 配置参考

作为存储管理员，您可以为 Ceph 对象网关设置各种选项。这些选项包含默认值。如果没有指定每个选项，则会自动设置默认值。

若要为这些选项设置特定值，可使用 ceph config set client.rgw OPTION VALUE 命令更新配置数据库。

A.1. 常规设置

名称	描述	类型	默认
rgw_data	设置 Ceph 对象网关的数据文件的位置。	字符串	**/var/lib/ceph/radosgw/$cluster-$id**
rgw_enable_apis	启用指定的 API。	字符串	**s3, s3website, swift, swift_auth, admin, sts, iam, notifications**
rgw_cache_enabled	是否启用 Ceph 对象网关缓存。	布尔值	**true**
rgw_cache_lru_size	Ceph 对象网关缓存中的条目数。	整数	**10000**
rgw_socket_path	域套接字的套接字路径。**FastCgiExternalServer** 使用这个套接字。如果不指定套接字路径，Ceph 对象网关将不会作为外部服务器运行。此处指定的路径必须与 **rgw.conf** 文件中指定的路径相同。	字符串	N/A
rgw_host	Ceph 对象网关实例的主机。可以是 IP 地址或主机名.	字符串	**0.0.0.0**
rgw_port	实例侦听请求的端口。如果未指定，Ceph 对象网关将运行外部 FastCGI。	字符串	无
rgw_dns_name	服务域的 DNS 名称。另请参阅 zone group 中的**主机名**设置。	字符串	无
rgw_script_uri	如果请求中没有设置 **SCRIPT_URI** 的备用值。	字符串	无

名称	描述	类型	默认
rgw_request_uri	**REQUEST_URI** 的备用值（如果请求中没有设置）。	字符串	无
rgw_print_continue	如果适用，启用 **100-continue**。	布尔值	**true**
rgw_remote_addr_param	远程地址参数.例如，如果反向代理正常运行，包含远程地址的 HTTP 字段或 **X-Forwarded-For** 地址。	字符串	**REMOTE_ADDR**
rgw_op_thread_timeout	打开线程的超时时间（以秒为单位）。	整数	600
rgw_op_thread_suicide_timeout	Ceph 对象网关进程结束前 **的超时时间**（以秒为单位）。如果设置为 **0**，则禁用。	整数	**0**
rgw_thread_pool_size	线程池的大小。	整数	512 个线程.
rgw_num_control_oids	用于不同 **rgw** 实例之间缓存同步的通知对象数量。	整数	**8**
rgw_init_timeout	Ceph 对象网关在初始化时放弃之前的秒数。	整数	**30**
rgw_mime_types_file	MIME 类型的路径和位置。用于 Swift 自动检测对象类型。	字符串	**/etc/mime.types**
rgw_gc_max_objs	在一个垃圾收集周期内可由垃圾回收处理的最大对象数量。	整数	**32**
rgw_gc_obj_min_wait	对象移除并通过垃圾回收处理处理前的最短等待时间。	整数	**2 * 3600**
rgw_gc_processor_max_time	两个连续垃圾回收处理周期开始之间的最长时间。	整数	**3600**
rgw_gc_processor_period	垃圾回收处理的周期时间。	整数	**3600**
rgw_s3 success_create_obj_status	**create-obj** 的替代成功状态响应。	整数	**0**
rgw_resolve_cname	**rgw** 是否应该使用请求主机名字段的 DNS CNAME 记录（如果主机名不等于 **rgw_dns name**）。	布尔值	**false**

名称	描述	类型	默认
rgw_object_stripe_size	Ceph 对象网关对象分条的大小。	整数	**4 << 20**
rgw_extended_http_attrs	添加一组可以在对象上设置的新属性。在放置对象时，可以通过 HTTP 标头字段设置这些额外属性。如果设置，这些属性将在对象上执行 GET/HEAD 时返回为 HTTP 字段。	字符串	none.例如："content_foo, content_bar"
rgw_exit_timeout_secs	在无条件退出之前等待进程的秒数。	整数	**120**
rgw_get_obj_window_size	单个对象请求的窗口大小，以字节为单位。	整数	**16 << 20**
rgw_get_obj_max_req_size	发送到 Ceph 存储群集的单个 get 操作的最大请求大小。	整数	**4 << 20**
rgw_relaxed_s3_bucket_names	为 zone group bucket 启用松散的 S3 存储桶规则。	布尔值	**false**
rgw_list buckets_max_chunk	列出用户 bucket 时，在单个操作中检索的最大 bucket 数量。	整数	**1000**
rgw_override_bucket_index_max_shards	bucket 索引对象的分片数量。值 **0** 表示没有分片。红帽不推荐设置太大的值（例如 **1000**），因为它会增加存储桶列表的成本。 该变量应在 **[client]** 或 **[global]** 部分中设置，以使其自动应用到 **radosgw-admin** 命令。	整数	**0**
rgw_curl_wait_timeout_ms	某些 **curl** 调用的超时时间（毫秒）。	整数	**1000**
rgw_copy_obj_progress	在长时间复制操作期间启用对象进度输出。	布尔值	**true**
rgw_copy_obj_progress_every_bytes	复制进度输出之间的最小字节数。	整数	**1024 * 1024**
rgw_admin_entry	管理请求 URL 的入口点。	字符串	**admin**

名称	描述	类型	默认
rgw_content_length_compat	启用 FCGI 请求的兼容性，同时设置了 CONTENT_LENGTH AND HTTP_CONTENT_LENGTH。	布尔值	**false**
rgw_bucket_default_quota_max_objects	每个 bucket 的默认最大对象数量。如果未指定任何其他配额，则在新用户上设置这个值。它对现有用户没有影响。 该变量应在 **[client]** 或 **[global]** 部分中设置，以使其自动应用到 **radosgw-admin** 命令。	整数	**-1**
rgw_bucket_quota_ttl	缓存的配额信息的时间（以秒为单位）是可信的。超时后，配额信息将从集群中重新获取。	整数	600
rgw_user_quota_bucket_sync_interval	在与集群同步前，收集存储桶配额信息的时间（以秒为单位）。在这段时间中，其他 RGW 实例不会看到来自此实例的操作的 bucket 配额统计的更改。	整数	180
rgw_user_quota_sync_interval	在同步集群前，会先累计用户配额信息的时间（以秒为单位）。在这段时间中，其他 RGW 实例不会看到此实例的操作中用户配额统计的更改。	整数	3600 * 24
log_meta	用来确定网关是否记录元数据操作的 zone 参数。	布尔值	**false**
log_data	用来确定网关是否记录数据操作的 zone 参数。	布尔值	**false**
sync_from_all	使用 **radosgw-admin** 命令设置或取消设置是否从所有区组对等进行区同步。	布尔值	**false**

A.2. 关于池

Ceph zone 映射到一系列 Ceph 存储群集池。

手动创建池与生成池

如果 Ceph 对象网关的用户密钥包含写入功能，则网关能够自动创建池。这对入门来说非常方便。但是，Ceph 对象存储群集使用 PG 默认值，除非已在 Ceph 配置文件中进行了设置。此外，Ceph 将使用默认的 CRUSH 层次结构。这些设置不是生产系统的理想选择。

Ceph 对象网关默认区的池包括：

- .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 对象网关基于每个区域创建池。如果手动创建池，请预先填充区域名称。系统池存储与对象相关的对象，如系统控制、日志记录和用户信息。按照惯例，这些池名称的前置为池名称的区域名称。

- .<zone-name>.rgw.control ：控制池。
- .<zone-name>.log ：日志池包含所有存储桶/容器的日志，以及对象操作的日志，如 create、read、update 和 delete。
- .<zone-name>.rgw.buckets.index ：此池存储存储桶的索引。
- .<zone-name>.rgw.buckets.data ：此池存储存储桶的数据。
- .<zone-name>.rgw.meta ：元数据池存储 user_keys 和其他关键元数据。

- *.<zone-name>.meta:users.uid ：用户 ID 池包含唯一用户 ID 的映射。*
- *.<zone-name>.meta:users.keys ：密钥池包含每个用户 ID 的访问密钥和 secret 密钥。*
- *.<zone-name>.meta:users.email ：电子邮件池包含与用户 ID 关联的电子邮件地址。*
- *.<zone-name>.meta:users.swift ：Swift 池包含用户 ID 的 Swift 子用户信息。*

Ceph 对象网关存储放置池中 bucket 索引(index_pool)和 bucket 数据(data_pool)的数据。它们可能会重叠；即，您可以对索引和数据使用相同的池。默认放置的索引池是 {zone-name}.rgw.buckets.index，默认放置的数据池是 {zone-name}.rgw.buckets。

名称	描述	类型	默认
rgw_zonegroup_root_pool	用于存储所有 zone group 特定信息的池。	字符串	**.rgw.root**
rgw_zone_root_pool	用于存储特定区域信息的池。	字符串	**.rgw.root**

A.3. 生命周期设置

作为存储管理员，您可以为 Ceph 对象网关设置各种 bucket 生命周期选项。这些选项包含默认值。如果没有指定每个选项，则会自动设置默认值。

若要为这些选项设置特定值，可使用 ceph config set client.rgw OPTION VALUE 命令更新配置数据库。

名称	描述	类型	默认
rgw_lc_debug_interval	对于开发人员，只能通过将到期规则从天数扩展到间隔（以秒为单位）来调试生命周期规则。红帽建议不要在生产环境中使用这个选项。	整数	**-1**
rgw_lc_lock_max_time	Ceph 对象网关内部使用的超时值。	整数	**90**

名称	描述	类型	默认
rgw_lc_max_objs	控制 RADOS 网关内部生命周期工作队列的分片，并且应仅设置为取消重新划分工作流的一部分。红帽建议不要在集群设置后更改此设置，而无需首先联系红帽支持。	整数	**32**
rgw_lc_max_rules	每个存储桶、生命周期配置文档中包含的生命周期规则数量。Amazon Web Service(AWS)限制为 1000 规则。	整数	**1000**
rgw_lc_max_worker	并行、处理存储桶和索引分片运行的生命周期 worker 线程数量。红帽不推荐在不联系红帽支持的情况下设置大于 10 的值。	整数	**3**
rgw_lc_max_wp_worker	每个生命周期 worker 线程可以并行处理的存储桶数量。红帽不推荐在不联系红帽支持的情况下设置大于 10 的值。	整数	**3**
rgw_lc_thread_delay	在几个点可以注入到分片处理中的延迟（以毫秒为单位）。默认值为 0。将值设为 10 到 100 ms 将减少 RADOS 网关实例的 CPU 使用率，并减少与观察到饱和度相关的生命周期线程工作负载容量的比例。	整数	**0**

A.4. SWIFT 设置

名称	描述	类型	默认
rgw_enforce_swift_acls	强制实施 Swift 访问控制列表(ACL)设置。	布尔值	**true**
rgw_swift_token_expiration	到期 Swift 令牌的时间（以秒为单位）。	整数	**24 * 3600**
rgw_swift_url	Ceph 对象网关 Swift API 的 URL。	字符串	无
rgw_swift_url_prefix	Swift API 的 URL 前缀，例如 **http://fqdn.com/swift**。	**swift**	N/A
rgw_swift_auth_url	用于验证 v1 身份验证令牌的默认 URL（如果不使用内部 Swift auth）。	字符串	无
rgw_swift_auth_entry	Swift 身份验证 URL 的入口点。	字符串	**auth**

A.5. 日志记录设置

名称	描述	类型	默认
debug_rgw_datacache	可以通过 **debug_rgw_datacache**子系统(**debug_rgw_datacache =30**)启用低级别 D3N 日志	整数	**1/5**
rgw_log_nonexistent_bucket	启用 Ceph 对象网关，以记录对不存在的 bucket 的请求。	布尔值	**false**
rgw_log_object_name	对象名称的日志格式。有关格式指定程序的详细信息，请参阅 man page 日期。	Date	**%Y-%m-%d-%H-%i-%n**
rgw_log_object_name_utc	记录的对象名称是否包含 UTC 时间。如果为 **false**，它将使用本地时间。	布尔值	**false**
rgw_usage_max_shards	使用日志记录的最大分片数量。	整数	**32**
rgw_usage_max_user_shards	用于单个用户使用日志记录的最大分片数量。	整数	**1**
rgw_enable_ops_log	为每个成功的 Ceph 对象网关操作启用日志记录。	布尔值	**false**
rgw_enable_usage_log	启用用量日志。	布尔值	**false**
rgw_ops_log_rados	操作日志是否应写入 Ceph 存储群集后端。	布尔值	**true**
rgw_ops_log_socket_path	用于编写操作日志的 Unix 域套接字。	字符串	无
rgw_ops_log_data-backlog	写入 Unix 域套接字的操作日志的最大数据大小。	整数	**5 << 20**
rgw_usage_log_flush_threshold	在异步清空前，用量日志中脏合并条目的数量。	整数	1024
rgw_usage_log_tick_interval	每 **n** 秒刷新待使用日志数据。	整数	**30**
rgw_intent_log_object_name	有意的日志对象名称的日志格式。有关格式指定程序的详细信息，请参阅 man page 日期。	Date	**%Y-%m-%d-%i-%n**
rgw_intent_log_object_name_utc	意图日志对象名称是否包含 UTC 时间。如果为 **false**，它将使用本地时间。	布尔值	**false**

名称	描述	类型	默认
rgw_data_log_window	数据日志条目窗口（以秒为单位）。	整数	**30**
rgw_data_log_changes_size	要保存的数据更改日志的内存中条目数量。	整数	**1000**
rgw_data_log_num_shards	要保留数据更改日志的分片（对象）的数量。	整数	**128**
rgw_data_log_obj_prefix	数据日志的对象名称前缀。	字符串	**data_log**
rgw_replica_log_obj_prefix	副本日志的对象名称前缀。	字符串	**replica log**
rgw_md_log_max_shards	元数据日志的最大分片数量。	整数	**64**
rgw_log_http_headers	以逗号分隔的 HTTP 标头列表，以包括在 ops 日志条目中。标头名称不区分大小写，并使用完整标头名称以及用下划线分隔的词语。	字符串	无

注意

不支持更改 rgw_data_log_num_shards 值。

A.6. KEYSTONE 设置

名称	描述	类型	默认
rgw_keystone_url	Keystone 服务器的 URL。	字符串	无
rgw_keystone_admin_token	Keystone admin 令牌（共享机密）。	字符串	无
rgw_keystone_accepted_roles	为请求提供服务所需的角色。	字符串	**Member, admin**
rgw_keystone_token_cache_size	每个 Keystone 令牌缓存中条目的最大数量。	整数	**10000**

A.7. KEYSTONE 集成配置选项

您可以将配置选项集成到 Keystone 中。有关可用 Keystone 集成配置选项的详细描述，请参见以下内容：

重要

在更新 Ceph 配置文件后，您必须将新的 Ceph 配置文件复制到存储集群中的所有 Ceph 节点。

rgw_s3_auth_use_keystone

描述

如果设置为 true，Ceph 对象网关将使用 Keystone 验证用户身份。

类型

布尔值

默认

false

nss_db_path

描述

NSS 数据库的路径。

类型

字符串

默认

""

rgw_keystone_url

描述

Keystone 服务器上管理 RESTful API 的 URL。

类型

字符串

默认

""

rgw_keystone_admin_token

描述

在 Keystone 中内部配置的令牌或共享机密用于管理请求。

类型

字符串

默认

""

rgw_keystone_admin_user

描述

keystone admin 用户名。

类型

字符串

默认

""

rgw_keystone_admin_password

描述

keystone admin 用户密码。

类型

字符串

默认

""

rgw_keystone_admin_tenant

描述

keystone v2.0 的 Keystone admin 用户租户。

类型

字符串

默认

""

rgw_keystone_admin_project

描述

keystone v3 的 keystone admin 用户项目。

类型

字符串

默认

""

rgw_trust_forwarded_https

描述

当 Ceph 对象网关前面的代理用于 SSL 终止时，它并不安全传入的 http 连接。启用这个选项来信任代理在决定连接安全时发送的转发和 X 转发的标头。这主要是服务器端加密所必需的。

类型

布尔值

默认

false

rgw_swift_account_in_url

描述

Swift 帐户是否在 URL 路径中编码。如果希望 Ceph 对象网关支持公开读取容器和临时 URL，则必须 将此选项设置为 true 并更新 Keystone 服务目录。

类型

布尔值

默认

false

rgw_keystone_admin_domain

描述

Keystone admin 用户域。

类型

字符串

默认

""

rgw_keystone_api_version

描述

要使用的 Keystone API 版本。有效选项为 2 或 3。

类型

整数

默认

2

rgw_keystone_accepted_roles

描述

为请求提供服务所需的角色。

类型

字符串

默认

成员、成员、管理员、

rgw_keystone_accepted_admin_roles

描述

允许用户获得管理特权的角色列表。

类型

字符串

默认

ResellerAdmin, swiftoperator

rgw_keystone_token_cache_size

描述

Keystone 令牌缓存中条目的最大数量。

类型

整数

默认

10000

rgw_keystone_verify_ssl

描述

如果为 true，Ceph 将尝试验证 Keystone 的 SSL 证书。

类型

布尔值

默认

true

rgw_keystone_implicit_tenants

描述

在他们自己具有相同名称的租户中创建新用户。在大多数情况下，将其设置为 true 或 false。为了与之前版本的 Red Hat Ceph Storage 兼容，也可以将其设置为 s3 或 swift。这会影响身份空间的影响，以便只有指定协议将使用隐式租户。一些较旧版本的 Red Hat Ceph Storage 仅支持具有 Swift 的隐式租户。

类型

字符串

默认

false

rgw_max_attr_name_len

描述

元数据名称的最大长度。0 跳过检查。

类型

大小

默认

0

rgw_max_attrs_num_in_req

描述

使用单个请求可以放置的最大元数据项目数。

类型

uint

默认

0

rgw_max_attr_size

描述

元数据值的最大长度。0 跳过检查

类型

大小

默认

0

rgw_swift_versioning_enabled

描述

启用 Swift 版本控制。

类型

布尔值

默认

0 或 1

rgw_keystone_accepted_reader_roles

描述

只能用于读取的角色列表。

类型

字符串

默认

""

rgw_swift_enforce_content_length

描述

在列出容器时发送内容长度

类型

字符串

默认

false`

A.8. LDAP 设置

名称	描述	类型	示例
rgw_ldap_uri	以 URI 格式的以空格分隔的 LDAP 服务器列表。	字符串	**ldaps://<ldap.your.domain>**
rgw_ldap_searchdn	LDAP 搜索域名，也称为基域。	字符串	**cn=users,cn=accounts,dc=example,dc=com**
rgw_ldap_binddn	网关将与此 LDAP 条目绑定（用户匹配）。	字符串	**uid=admin,cn=users,dc=example,dc=com**
rgw_ldap_secret	包含 **rgw_ldap_binddn** 凭据的文件。	字符串	**/etc/openldap/secret**
rgw_ldap_dnattr	包含 Ceph 对象网关用户名的 LDAP 属性（以构成 binddns）。	字符串	**uid**