Red Hat Ceph Storage 7

边缘指南

Red Hat Ceph Storage 的 Edge 集群指南

Last Updated: 2024-06-14

Red Hat Ceph Storage 7 边缘指南

Red Hat Ceph Storage 的 Edge 集群指南

法律通告

摘要

本文档提供有关 Edge 集群的信息，这是具有成本效益对象存储配置的解决方案。红帽承诺替换我们的代码、文档和网页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 边缘集群

边缘集群是具有成本效益的对象存储配置的解决方案。

红帽支持 Red Hat Ceph Storage 集群的以下最低配置：

- 三个用于 SSD 的副本的节点集群。
- 为 HDD 有三个副本的四节点集群。
- 具有 2+2 配置的 EC 池的四节点集群。

使用较小的集群时，利用率会停机，因为使用量量和弹性的损失。

第 2 章 池概述

Ceph 客户端将数据存储在池中。创建池时，您要为客户端创建一个 I/O 接口来存储数据。

从 Ceph 客户端的角度来看，即块设备、网关和其他，与 Ceph 存储集群交互非常简单：

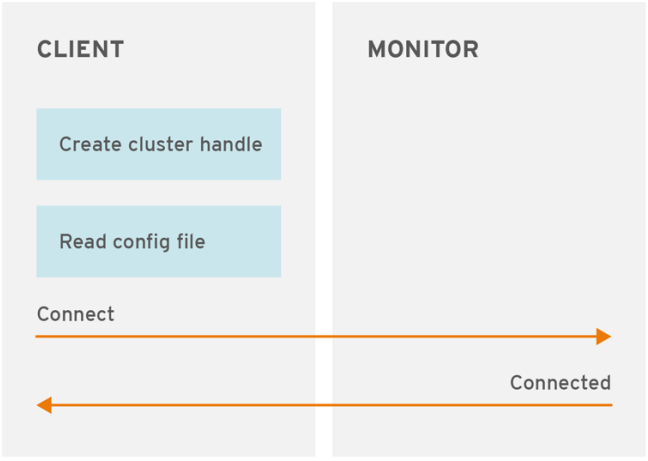
- 创建集群句柄。
- 将集群句柄连接到集群。
- 创建用于读取和写入对象及其扩展属性的 I/O 上下文。

创建集群句柄并连接到集群

要连接到 Ceph 存储集群，Ceph 客户端需要以下详情：

- 集群名称（默认为 Ceph ）- 不使用通常，因为它听起来很不确定。
- 初始监控地址。

Ceph 客户端通常使用 Ceph 配置文件的默认路径检索这些参数，然后从文件读取，但用户也可以在命令行中指定参数。Ceph 客户端还提供用户名和密钥，身份验证默认为 **on**。然后，客户端联系 Ceph 监控集群，并检索 cluster map 的最新副本，包括其 monitor、OSD 和池。



创建池 I/O 上下文

为了读取和写入数据，Ceph 客户端会为 Ceph 存储集群中的特定池创建一个 I/O 上下文。如果指定的用户具有池的权限，Ceph 客户端可以从指定的池读取和写入。

Ceph 的架构使存储集群能够将此简单接口提供给 Ceph 客户端，以便客户端可以通过指定池名称并创建 I/O 上下文来选择您定义的一个复杂的存储策略。存储策略对 Ceph 客户端在容量和性能方面不可见。类似地，Ceph 客户端的复杂性（如将对象映射到块设备表示或提供 S3/Swift RESTful 服务）对 Ceph 存储集群不可见。

池为您提供了弹性、放置组、CRUSH 规则和配额。

- **弹性** ：您可以设置允许多少个 OSD 失败，而不丢失数据。对于复制池，这是对象的所需副本数。典型的配置存储一个对象和一个额外副本，即 **size = 2**，但您可以确定副本或副本数。对于纠删代码池，它是 **纠删代码** profile 中的编码区块数，即 **m=2**。

- **放置组** ：您可以为池设置放置组数量。典型的配置为每个 OSD 使用大约 50 到 100 个放置组来提供最佳平衡，而无需使用太多计算资源。在设置多个池时，要小心，请务必为池和集群设置合理的放置组数量。

- **CRUSH 规则** ：当您在池中存储数据时，映射到池的 CRUSH 规则可让 CRUSH 识别每个对象及其副本放置的规则，或者集群中纠删代码池的块。您可以为池创建自定义 CRUSH 规则。

- **配额** ：当您使用 **ceph osd pool set-quota** 命令在池中设置配额时，您可以限制对象的最大数量或指定池中存储的最大字节数。

第 3 章 弹性和非弹性数据池

什么是弹性池和非弹性池？

弹性数据池支持数据复制或编码其数据。

非弹性数据池不会启用复制或编码其数据。

非弹性池也称为 **replica1**，不受数据丢失的影响。具有非弹性数据池的小型集群实现自己的复制，不需要从 Red Hat Ceph Storage 复制，因为它执行自己的复制。

数据池的集群利用率

使用较小的配置时，集群利用率会因为弹性丢失而减少。恢复受主机利用率的限制，可能会影响每秒的生产输入/输出操作(IOPS)。当只有一个故障节点时，您可以添加第三个节点来限制主机利用率和 Red Hat Ceph Storage，以恢复完整复制。

第 4 章 CEPH 纠删代码

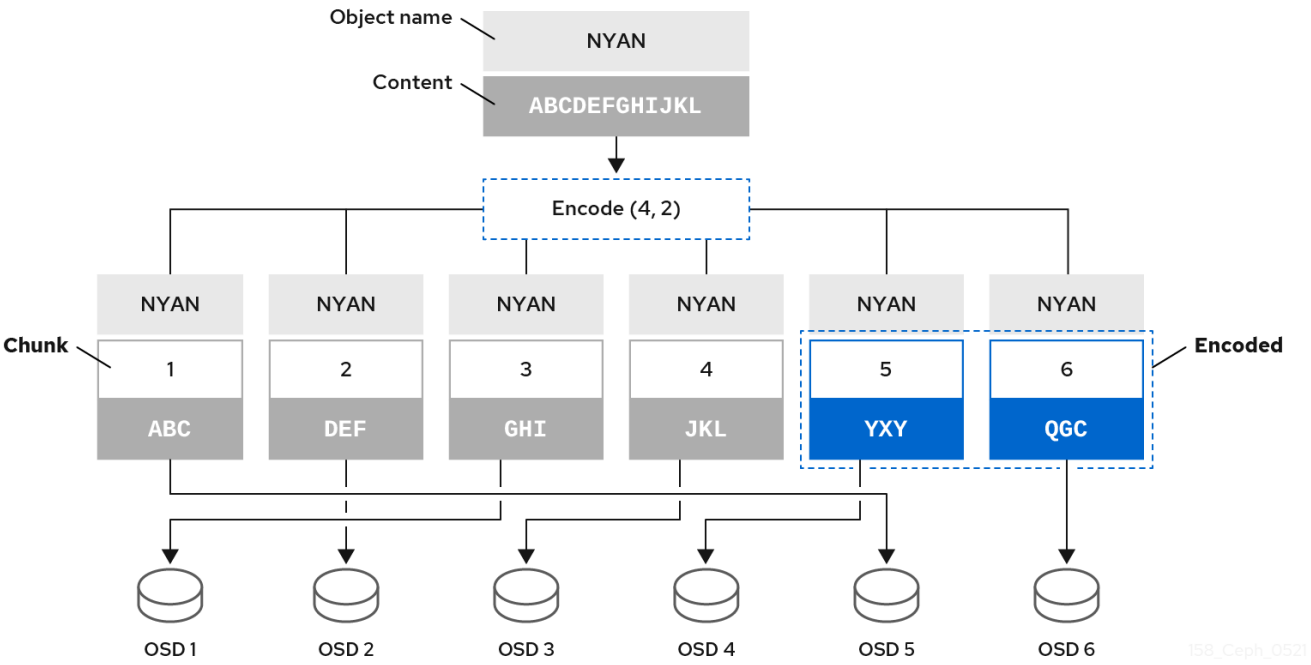
Ceph 可以加载许多纠删代码算法之一。最早且最常用的是 **Reed-Solomon** 算法。纠删代码实际上是一个转发错误修正(FEC)代码。FEC 代码会将 **K** 块的消息转换为较长的消息，称为"代码字"的 **N** 块，以便 Ceph 可以从 **N** 块的子集恢复原始消息。

更具体地说，**N = K+M**，其中变量 **K** 是原始数据块的数量。变量 **M** 代表额外的或者冗余区块，它添加了纠删代码算法，以提供防故障的保护。变量 **N** 是纠删代码池进程后创建的区块总数。**M** 的值是 **N-K**，这意味着算法计算来自 **K** 原始数据块的 **N-K** 冗余块。此方法可确保 Ceph 可以访问所有原始数据。系统可应对任意 **N-K** 失败。例如，在 10 **K** 中是 16 **N** 配置，或者纠删代码 **10/16**，纠删代码算法向 10 个基本区块 **K** 增加了六个额外的区块。例如，在 **M = K-N** 或 **16-10 = 6** 配置中，Ceph 会将 16 个块 **N** 分布到 16 个 OSD 中。即使有 6 个 OSD 无法丢失数据，也可以从 10 个验证的 **N** 块重新创建原始文件，从而确保非常高的容错能力。

与复制池一样，在纠删代码池中，设置中的 Primary OSD 会接收所有写入操作。在复制池中，Ceph 在集合的次要 OSD 上的 PG 中对 PG 中的各个对象进行深度副本。对于纠删代码，进程略有不同。纠删代码池将每个对象存储为 **K+M** 块。它被分为 **K** 数据区块和 **M** 编码区块。该池配置为具有 **K+M** 大小，以便 Ceph 将每个块存储到操作集的 OSD 中。Ceph 将块的排名存储为对象的属性。Primary OSD 负责将载荷编码到 **K+M** 区块，并将它们发送到其他 OSD。Primary OSD 还负责维护 PG 日志的权威版本。

例如，在典型的配置中，系统管理员会创建一个纠删代码池以使用六个 OSD 并保持丢失两个 OSD。也就是说，(**K+M = 6**)， (**M = 2**)。

当 Ceph 将包含 **ABCDEFGHIJKL** 的对象 **NYAN** 写入池时，通过简单地将内容划分为四个部分，即 **ABC**、**DEF**、**GHI** 和 **JKL**。如果内容长度不是 **K** 的倍数，则该算法会对内容进行 pad 处理。此函数还会创建两个编码块：第五个带有 **YXY**，以及第六个带有 **QGC** 的编码块。Ceph 将 OSD 上的各个块存储在执行集合中，其中将区块存储到具有相同名称 **NYAN** 但驻留于不同的 OSD 的对象中。除了其名称外，该算法还必须保留创建块作为对象 **shard_t** 的属性的顺序。例如，Chunk 1 包含 **ABC** 和 Ceph 存储在 **OSD5** 上，而块 5 包含 **YXY**，Ceph 则存储在 **OSD4** 上。



在恢复场景中，客户端尝试通过读取块 1 到 6 的块从纠删代码池中读取对象 **NYAN**。OSD 告知算法缺少块 2 和 6。这些缺少的块被称为 'erasures'。例如，由于 OSD6 不足而无法读取块 6，因此 **OSD6** 无法读取块 2，因为 **OSD2** 是最慢的，并且其块没有考虑。但是，当算法有四个块时，它会读取四个块：块 1，包含 **ABC**、块 3 包含 **GHI**、块 4 包含 **JKL**，以及包含 **YXY** 的块 5。然后，它会重建对象 **ABCDEFGHIJKL** 的原始内容，以及包含 **QGC** 的块 6 的原始内容。

将数据分割为块独立于对象放置。CRUSH 规则集和纠删代码池配置文件决定了 OSD 上的区块放置。例如，在纠删代码配置集中使用 Locally Repairable Code(**lrc**)插件会创建额外的块，需要较少的 OSD 从中恢复。例如，在 **lrc** 配置集配置 **K=4 M=2 L=3** 中，该算法会创建六个块(**K+M**)，就像 **jerasure** 插件一样，但本地的特性值(**L=3**)要求算法在本地创建 2 个块。该算法会创建额外的块，例如 **(K+M)/L**。如果包含块 0 的 OSD 失败，可以使用块 1、2 和第一个本地块来恢复此块。在这种情况下，算法只需要 3 个块而不是 5 个就能恢复。

注意

使用纠删代码的池禁用 Object Map。

重要

对于带有 2+2 配置的纠删代码池，请将 **ABCDEFGHIJKL** 的输入字符串替换为 **ABCDEF**，并将编码区块从 **4** 替换为 **2**。

其它资源

- 有关 CRUSH、纠删代码 profile 和插件的更多信息，请参见 Red Hat Ceph Storage 7 的存储策略指南。
- 如需有关 Object Map 的更多信息，请参阅 *Ceph 客户端对象映射*部分。

第 5 章 纠删代码池概述

Ceph 默认使用复制池，这意味着 Ceph 将每个对象从 Primary OSD 节点复制到一个或多个次要 OSD。纠删代码（erasure-coded）池可以减少确保数据持久性所需的磁盘空间量，但它的计算比复制要高得多。

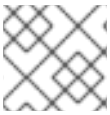
Ceph 存储策略涉及定义数据持久性要求。数据持久性意味着，在丢失一个或多个 OSD 时可以保持数据不会丢失。

Ceph 在池中存储数据，其中有两种池类型：

- *复制*
- *erasure-coded*

纠删代码是一种在 Ceph 存储群集中存储对象的方法，即：该算法将对象分割为数据区块(**k**)和编码区块(**m**)，并将这些区块存储在不同的 OSD 中。

如果 OSD 失败，Ceph 会从其他 OSD 检索剩余的数据(**k**)和编码(**m**)块，以及纠删代码算法从这些块中恢复对象。



注意

红帽建议 **min_size** 用于纠删代码池为 **K+1** 或更高版本，以防止丢失写入和数据。

纠删代码使用存储容量比复制更高效。n-replication 方法维护对象的 **n** 个副本（在 Ceph 中默认为 3x），而区块则仅维护 **k** + **m** 个区块。例如，3 个数据和 2 个编码区块使用 1.5 倍的原始对象的存储空间。

纠删代码使用比复制少的存储开销，但纠删代码池使用的内存比访问或恢复对象时使用的 CPU 数要多。当数据存储需要具有持久性和容错能力，但不需要快速读取性能（如冷存储、历史记录等）时，擦除代码具有优势。

有关纠删代码如何在 Ceph 中工作的信息，请参阅 Red Hat Ceph Storage 7 架构指南中的 *Ceph Erasure Coding* 部分。

*Ceph 在使用 **k=2** 和 **m=2** 初始化集群时 **会创建一个默认** 纠删代码配置集，这意味着 Ceph 将通过三个 OSD (**k+m == 4**)分散对象数据，Ceph 可以在不丢失数据的情况下丢失其中一个 OSD。要了解有关纠删代码性能分析的更多信息，请参阅 Erasure Code Profiles 部分。*

重要

*将 **.rgw.buckets** 池配置为纠删代码，所有其他 Ceph 对象网关池都复制，否则尝试创建新存储桶失败，并显示以下错误：*

```
set_req_state_err err_no=95 resorting to 500
```

*这样做的原因是，纠删代码池不支持 **omap** 操作，某些 Ceph Object Gateway 元数据池需要 **omap** 支持。*

5.1. 创建纠删代码池示例

*创建纠删代码池并指定放置组。**ceph osd pool create** 命令会创建一个带有 default 配置集的纠删代码池，除非指定了另一个配置集。配置集通过设置两个参数 **k** 和 **m** 来定义数据的冗余性。这些参数定义数据被分割的区块数量，并创建编码区块数量。*

最简单的纠删代码池等同于 RAID5，且至少需要四个主机。您可以创建一个带有 2+2 配置集的纠删代码池。

流程

1. 在带有 2+2 配置的四个节点上为纠删代码池设置以下配置。

   语法

   ```
   ceph config set mon mon_osd_down_out_subtree_limit host
   ceph config set osd osd_async_recovery_min_cost 1099511627776
   ```

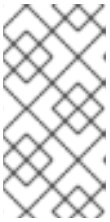
   重要

   通常不需要纠删代码池。

   重要

   async 恢复成本是副本后面的 PG 日志条目数以及缺失的对象数量。**osd_target_pg_log_entries_per_osd** 是 **30000**。因此，具有单个 PG 的 OSD 可能具有 **30000** 条目。由于 **osd_async_recovery_min_cost** 是 64 位整数，因此将带有 2+2 配置的 EC 池的 **osd_async_recovery_min_cost** 的值设置为 **1099511627776**。

   注意

   对于具有四个节点的 EC 集群，K+M 的值为 2+2。如果节点完全失败，它不会恢复为四个块，且只有三个节点可用。当您将 **mon_osd_down_out_subtree_limit** 的值设置为 **host** 时，它会在主机停机场景中阻止 OSD 标记为 out，因此防止数据重新平衡，并等待节点再次启动。

2. 对于带有 2+2 配置的纠删代码池，请设置配置集。

   语法

   ```
   ceph osd erasure-code-profile set ec22 k=2 m=2 crush-failure-domain=host
   ```

   Example

   ```
   [ceph: root@host01 /]# ceph osd erasure-code-profile set ec22 k=2 m=2 crush-failure-domain=host

   Pool : ceph osd pool create test-ec-22 erasure ec22
   ```

3. 创建纠删代码池。

   Example

   ```
   [ceph: root@host01 /]# ceph osd pool create ecpool 32 32 erasure

   pool 'ecpool' created
   ```

```
$ echo ABCDEFGHI | rados --pool ecpool put NYAN -
$ rados --pool ecpool get NYAN -
ABCDEFGHI
```

32 是放置组的数量。

第 6 章 后端压缩

使用压缩选项压缩较小的容量的边缘集群。

BlueStore 允许两种类型的压缩：

- 常规工作负载的 BlueStore 压缩级别。
- S3 工作负载的 Ceph 对象网关压缩级别。

有关压缩算法的更多信息，请参阅 池值。

您需要启用压缩，并在池中启用压缩时确保集群中不会崩溃。

您可以使用以下方法在边缘集群池中启用压缩：

- 启用支持的压缩算法，如 snappy、zlib 和 zstd，并使用以下命令启用支持的压缩模式，如 **None**、**被动**、active 和 **force** ：

  语法

  ```
  ceph osd pool set POOL_NAME compression_algorithm ALGORITHM
  ceph osd pool set POOL_NAME compression_mode MODE
  ```

- 使用以下命令启用各种压缩率：

  语法

  ```
  ceph osd pool set POOL_NAME compression_required_ratio RATIO
  ceph osd pool set POOL_NAME compression_min_blob_size SIZE
  ceph osd pool set POOL_NAME compression_max_blob_size SIZE
  ```

- 创建三个池并在这些池中启用不同的压缩，以确保池中不会发生 IO 停止页面。

- 创建第四个池，但不在池中创建任何压缩。使用压缩写入与池相同的数据。压缩的池使用较少的 RAW 空间，池无需压缩。

要验证 已设置了这些算法，请使用 ceph osd pool get POOL_NAME OPTION_NAME 命令。

要 取消设置 这些算法，请使用 ceph osd pool unset POOL_NAME OPTION_NAME 命令以及适当的选项。

第 7 章 集群拓扑和共处

了解所需的拓扑，以及边缘集群的并置需要考虑的因素。

有关集群拓扑、与 OpenStack 超融合、OpenStack 上的节点协调以及 OpenStack 最小配置的限制的信息，请参阅 HCI 的 Ceph 配置覆盖。

有关 colocation 的更多信息，请参阅 共处。