

Ceph介绍



概述

- Ceph是Sage Weil 在 UCSC开发的大规模的、可扩展的、开源的分布式存储系统
- Ceph client included in Linux kernel since 2.6.34
- Supported by Openstack since Folsom
- Maintained by Inktank inc.





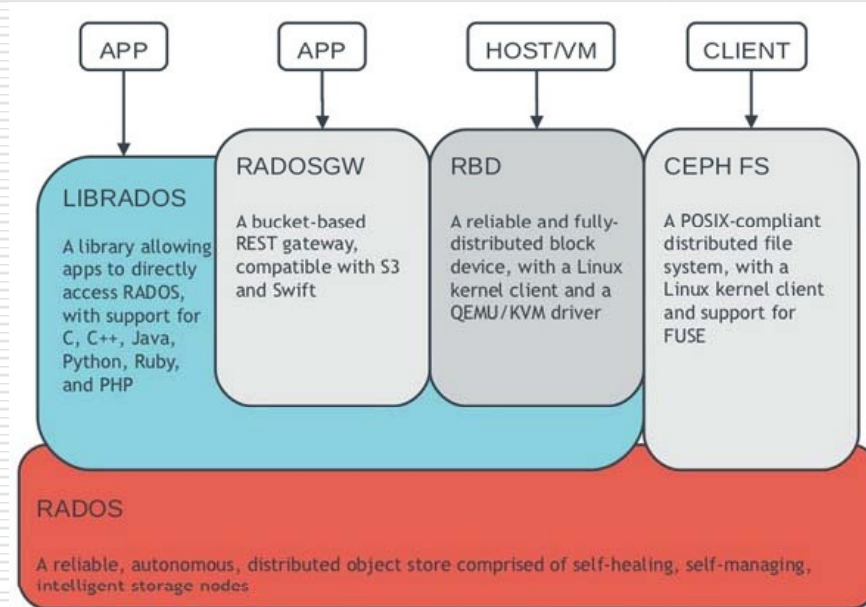
概述

RADOS

- ◆ Ceph的核心组件
- ◆ 提供高可靠、高可扩展的分布式对象存储架构
- ◆ 利用本地文件系统存储对象

Client

- ◆ Cephfs
- ◆ RBD
- ◆ Radosgw
- ◆ Librados





Client

■ Cephfs

- ◆ 提供POSIX兼容的分布式文件系统接口，支持 kernel、fuse

■ RBD

- ◆ 提供高可靠的，分布式的块接口

■ Radosgw

- ◆ 提供S3、Swift兼容的对象接口

■ Librados

- ◆ 提供rados库调用接口



Rados特色

■ 高可靠性

- ◆ 多副本
- ◆ 自动隔离失效节点
- ◆ 数据自动恢复

■ 高可扩展性

- ◆ 数据分布式存储
- ◆ 数据透明扩容

■ CRUSH数据分布算法





Ceph for openstack

■ rbd

- ◆ 为Glance、Cinder提供镜像存储
- ◆ 提供Qemu/KVM驱动支持
- ◆ 支持openstack的虚拟机迁移

■ rgw

- ◆ 替代swift

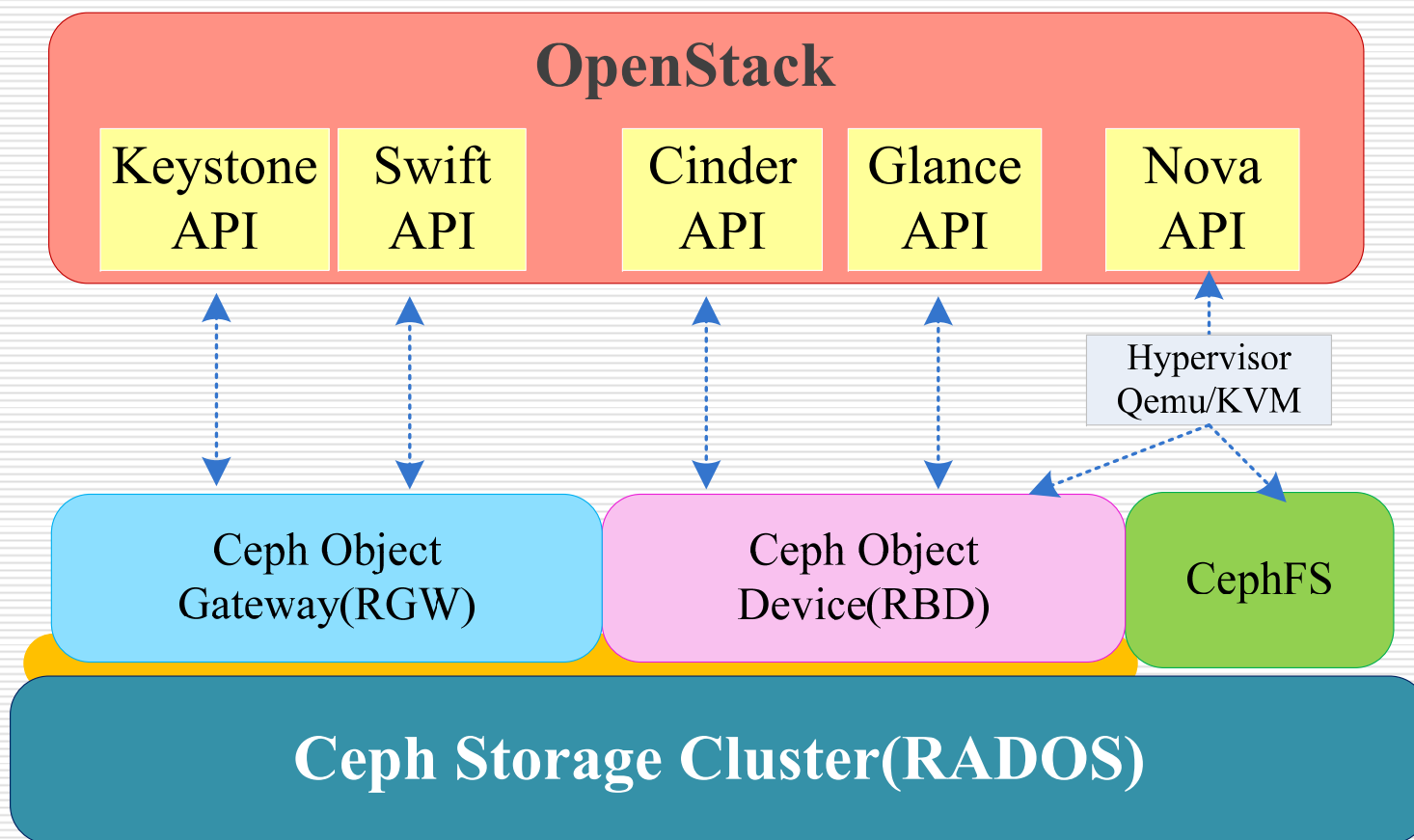
■ CephFS

- ◆ 提供共享的文件系统存储
- ◆ 支持openstack的虚拟机迁移

■ 还可替代hdfs



Ceph for Openstack





Cephfs组成

■ OSD

- ◆ 存储文件数据和元数据

■ Monitor

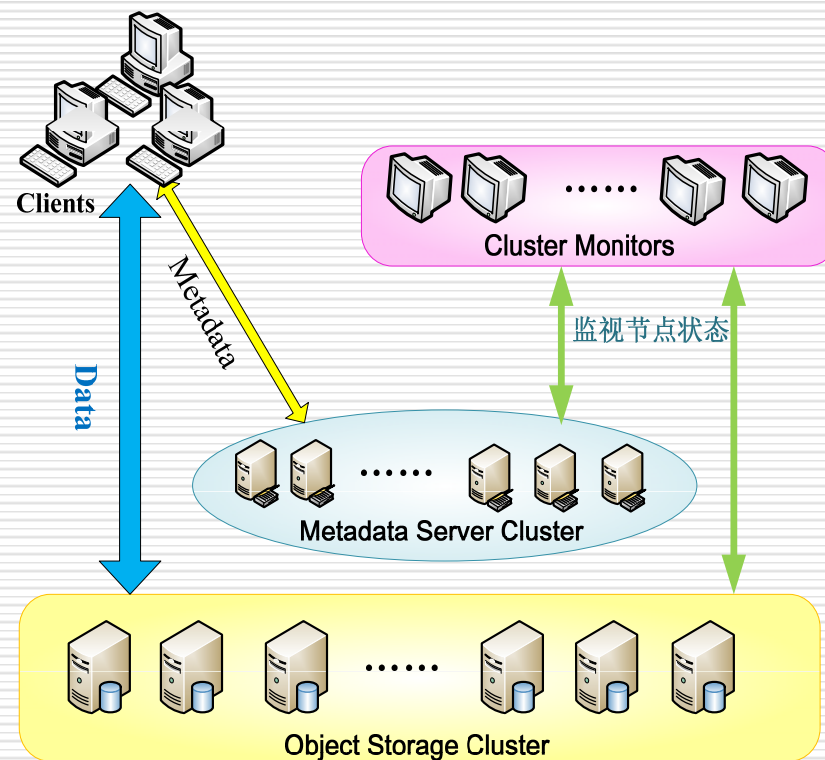
- ◆ 监视整个集群状态
- ◆ 维护并通告集群拓扑结构

■ MDS

- ◆ 缓存和同步元数据
- ◆ 管理名字空间

■ Client

- ◆ 提供POSIX接口





Cephfs特色

■ 数据与元数据分离

- ◆ 数据由OSD服务

- ◆ 元数据由MDS服务

- ◆ 但都存储在OSD上，作为不同的对象，MDS作为元数据cache

■ 文件分片存储，支持条带存储

■ OSD、Monitor、MDS均支持集群运行，支持动态增删节点，无单点失效

■ 动态子树划分



capability

- 为了维护数据一致性, 由mds给client颁发file cap, 决定client允许进行的操作
- Fb: 允许client进行带缓存的写操作
- Fc: 允许client进行带缓存的读操作
- 对于同一个文件, 当有大于1个的writer或1个writer和1个或多个reader时, client将进行无缓存的同步读写, 除非打开LAZYIO



Ceph应用

- Ceph遵循LGPL协议
- Ceph支持的云计算和大数据平台：
OpenStack、CloudStack、OpenNebula、Hadoop等
- Ceph合作伙伴：SUSE、Citrix、Canonical、DELL等

apachecloudstack™



openstack™



CANONICAL

CITRIX



SUSE



The power to do more



我们的工作

- **Inline data support**
- **Fallocate support**
- **Ceph测评**



我们的工作

- **Inline data support**

- Fallocate support

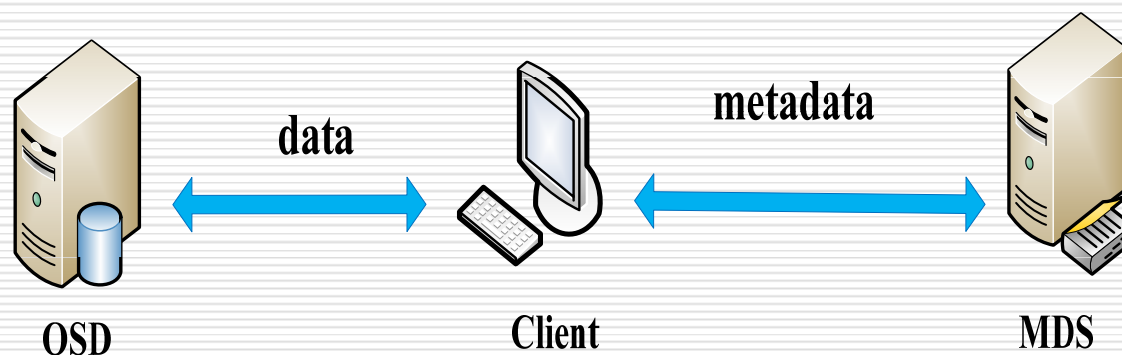
- Ceph测评



3.1 Inline data support

■ 动机

- ◆ 元数据与数据分离存储
- ◆ 文件访问需要两次通信：从MDS取元数据、从OSD取数据
- ◆ 小文件访问受限于网络延迟





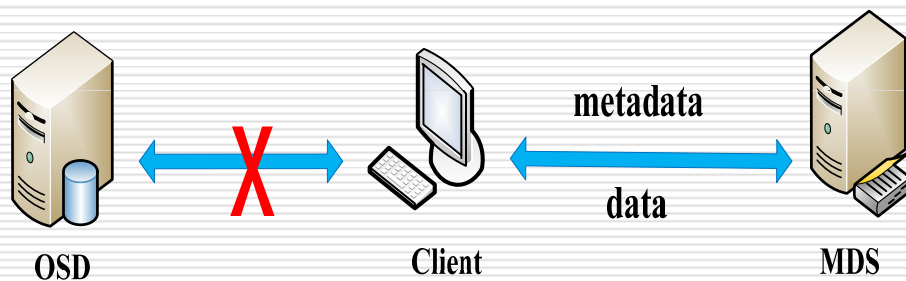
3.1 Inline data support

方法

- ◆ 针对小文件，数据存储在元数据中
- ◆ 打开文件时，元数据服务器将数据与元数据同时推送给客户端

优势

- ◆ 节省了数据存储位置计算时间
- ◆ 节省了与OSD的通信





Inline data support

■ 文件状态设计

- ◆ Inline 状态：文件所有数据都存放在元数据区域，OSD上不存数据
- ◆ Disable状态：文件所有数据从元数据区域完全迁移至OSD，元数据区域不再存放文件数据

■ 状态变迁

- ◆ 记录并维护inline data版本，记录inline状态
- ◆ 状态只能从inline状态变迁到disable状态，不可逆转



Inline data support

■ 状态迁移条件

◆ 文件大小超过阈值

◆ 当client无法执行带缓存的read, write, 而只能同步读写数据时

■ 典型情况: 1 writer + 1 or more readers 或 2 or more writers

■ 为保证数据一致性, 此时ceph取消client buffer, 强制client进行无page cache的同步读写

■ 继续inline对mds负载较重

■ 迁移处理比较复杂, 因为client不定拥有完整且最新的inline data



Some tricks

- inline data迁移异步进行
 - 避免反复推送uptodate inline data
 - 避免multiple clients同时迁移inline data,导致修改丢失
 - ◆ Client 1 打开文件,写入了区域[a, b], 其中 $a < \text{threshold}$, $b > \text{threshold}$, 1将inline data迁移到了osd, 注意此时[a, threshold]中包含了1修改后的数据, 1还没有通知mds inline data已撤销
 - ◆ Client 2打开文件,导致cap被撤销,发现没FB,迁移数据,这样旧的inline data将覆盖1的修改
 - ◆ 为第1个对象引入一个扩展属性记录数据版本, client迁移数据时,先比较该扩展属性,如果该扩展属性不存在或client的版本高于该扩展属性的值时,才能写入
 - 避免一个client迁移inline data成功,但在未通知mds之前失效,以后新的inline data不能覆盖osd上旧的inline data
-



Inline data support

■ 数据结构设计

◆ inode添加两个域

- uint64_t inline_version

- bufferlist inline_data

◆ inline_version

- 标识inline data的版本

- 维护数据的一致性和正确性

- 从1开始单调递增，最大值为 $2^{64} - 1$ ，即
CEPH_INLINE_DISABLED，定义为inline data support
的disabled状态

◆ inline_data

- 存放小文件的实际文件内容



文件操作流程

■ Create

- ◆ 初始化inline_data及inline_version
- ◆ Client: inline_version=0
- ◆ MDS: inline_version=1

■ Open

- ◆ 元数据查询: getattr
- ◆ MDS向Client发送元数据
- ◆ 为客户端分配操作文件的capability, 如FC、FB
权限



文件执行流程

■ Read

- ◆ 在客户端inline version < CEPH_INLINE_DISABLED时，表示是inline data状态
- ◆ 判断客户端是否有读缓存的权限
 - 如果有，返回inline data
 - 如果没有，则将inline data迁移到OSD，执行原read流程，再将inline version设置为CEPH_INLINE_DISABLED



文件执行流程

Write

- ◆ 在 Client inline_version < CEPH_INLINE_DISABLED 时，表示是 inline data 状态
- ◆ 判断 Client 是否有 FB 权限
 - 如果有 FB 权限，判断写文件范围是否超出设置的 inline 阈值
 - ◆ 如果未超出，则将文件数据写入到 inline data，inline_version+1, mark caps dirty, 异步将 inline data 写到 mds
 - ◆ 如果超出，执行如下步骤：
 - ▶ 将 inline data 迁移至 OSD
 - ▶ 执行原有写操作写 OSD
 - ▶ 迁移完成后，将 Inode 中的 inline data 清空，将 inline_version 置为 CEPH_INLINE_DISABLED
 - 如果没有权限，执行超出 inline 阈值的相同流程



我们的工作

■ Inline data support

■ **Fallocate support**

■ Ceph测评



fallocate support

■ 原型:

◆ `int fallocate(int fd, int mode, off_t offset, off_t len)`

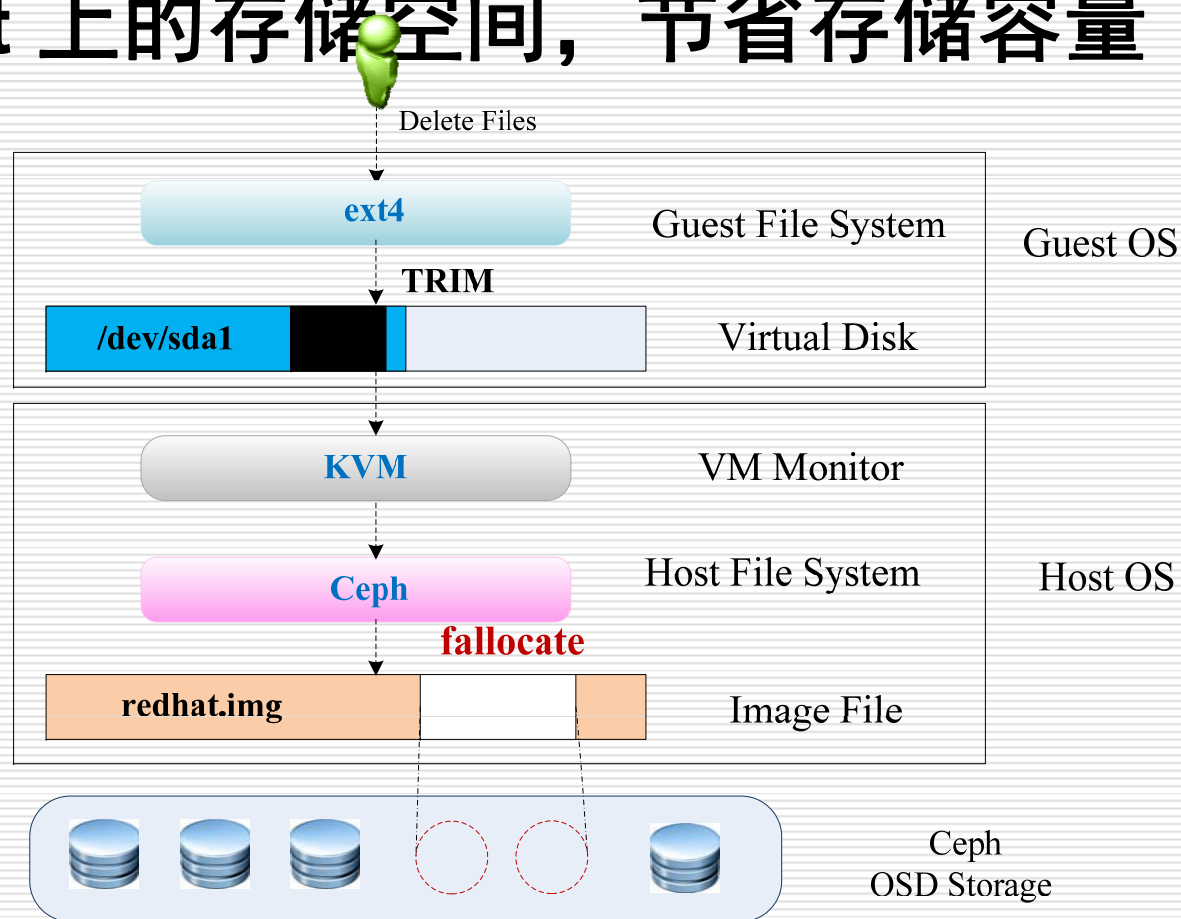
■ 功能

- ◆ 为文件预分配存储空间: 从offset开始预分配大小为len的空间
- ◆ `FALLOC_FL_PUNCH_HOLE`: 释放磁盘空间



fallocate用途

- Ceph用作虚拟机镜像文件存储时，可以释放host 上的存储空间，节省存储容量





算法

■ OSD object处理

- ◆ 当hole覆盖了整个object时,将该object从osd上删除
- ◆ 如果hole只涉及object的一部分,将该部分写0

■ Page cache

- ◆ 如果hole覆盖page cache中的整个page,将该page写回,删除
- ◆ 如果hole只涉及page的一部分,将该部分写0



Some tricks

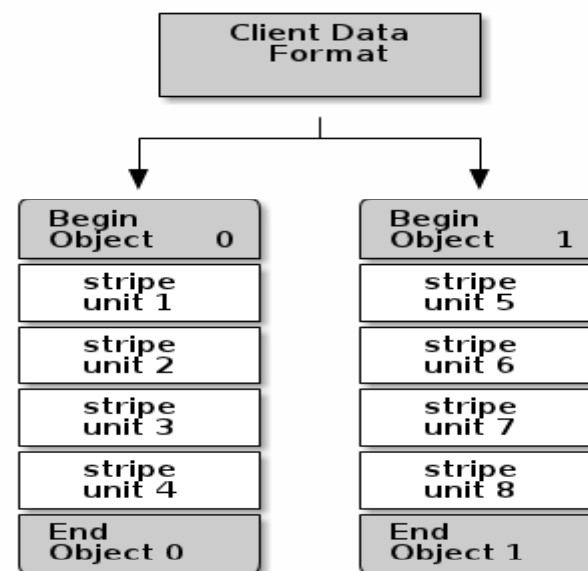
- 如果hole覆盖第一个object,由于其包含有backtrace等元数据信息,不将其删除,而是将其truncate到0

- Data striping支持

- ◆ 把连续的数据分割成相同大小的数据块,把每段数据分别写入到不同对象上的方法

- ◆ 参数

- Object Size
 - Stripe Width
 - Stripe Count





Ceph评测

■ 测试环境：

硬件环境		软件环境	
CPU	intel 至强 2路12核	操作系统	ubuntu 12.04
内存	64GB	Ceph版本	0.56.6
网卡	万兆		
硬盘	200GB		

■ 测试内容：

◆ 文件系统的顺序读写、随机读写性能

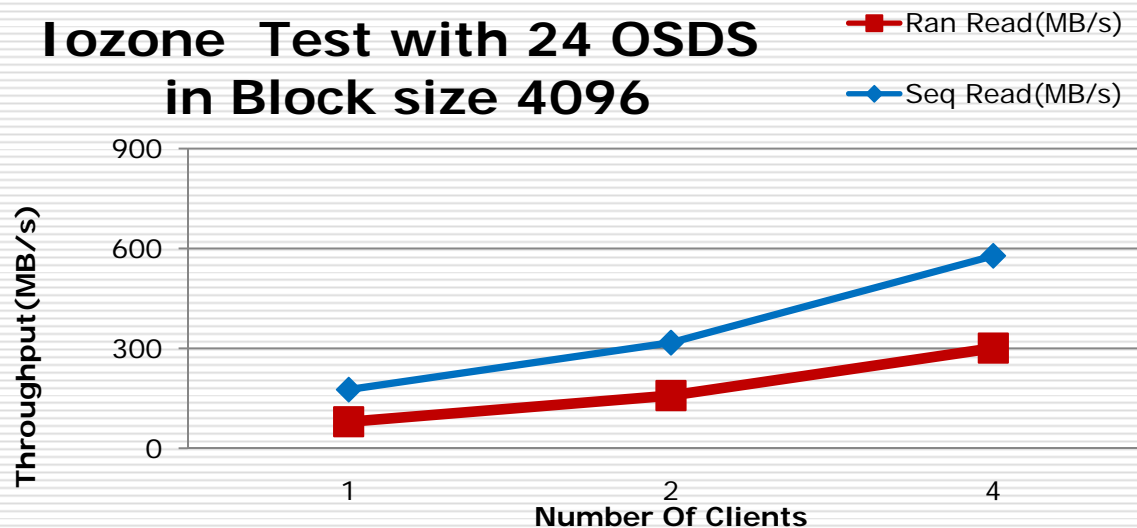
■ 测试工具：

◆ lozone



Ceph评测

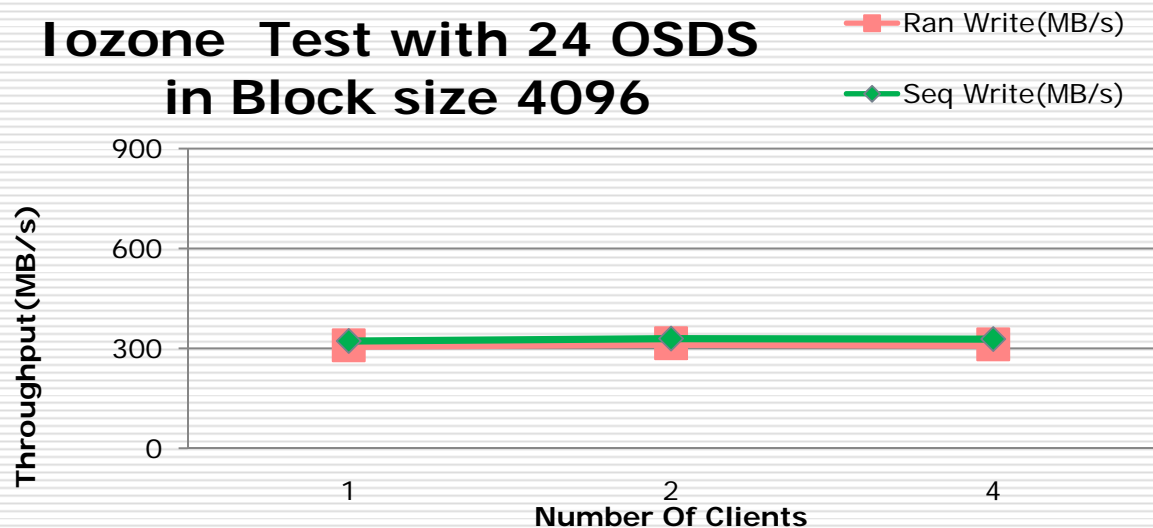
- 固定OSD、Block Size时，Client数增加各项性能变化曲线如下：





Ceph评测

- 固定OSD、Block Size时，Client数增加各项性能变化曲线如下：





评测分析

■ 条件

- ◆ 固定OSD个数（24个），固定Block Size（4MB）

■ 结论

- ◆ 随着client数量的增加，写性能（包括顺序写和随机写）基本不变
- ◆ 随着client数量的增加，读性能（包括顺序读、随机读和跳读）呈线性增长

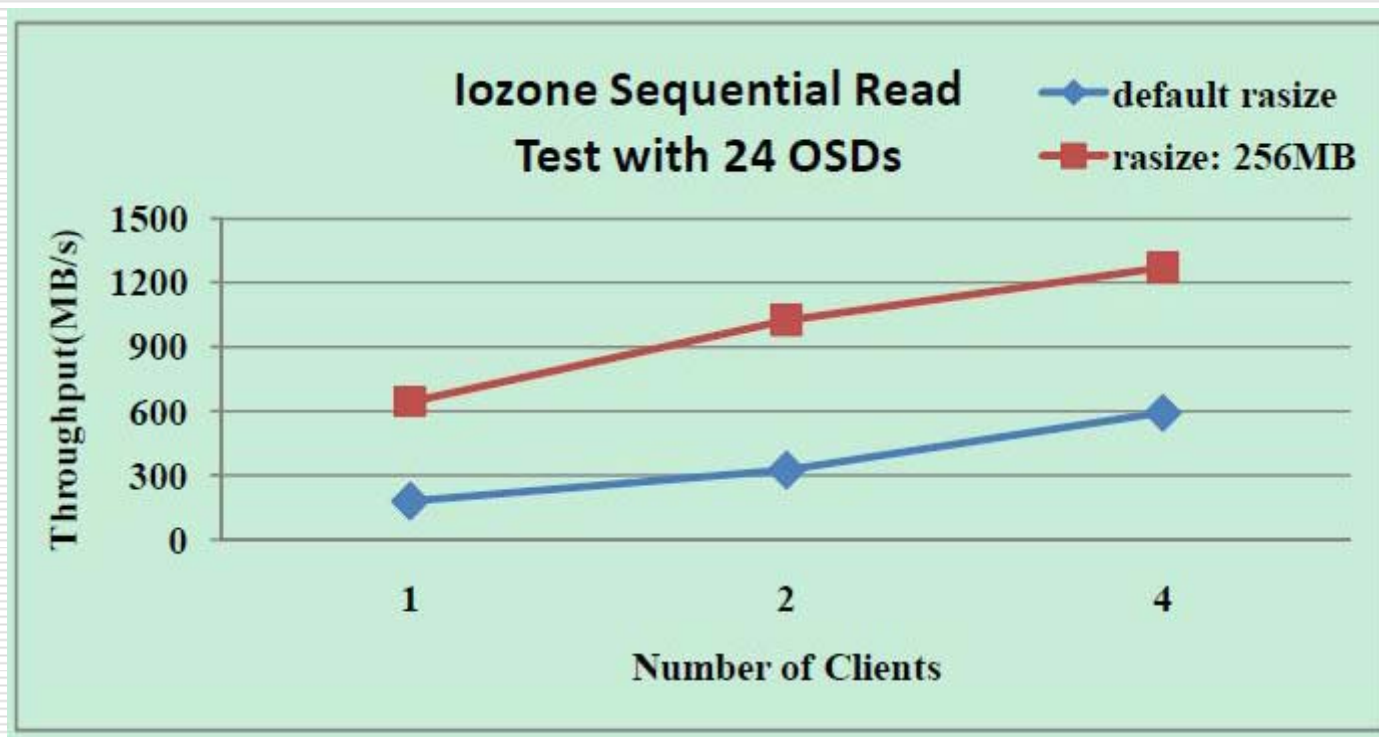
■ 分析

- ◆ 写操作时，OSD向磁盘写入速度有限，1个Client对Server端的压力已经足够，因此，增加Client，写速度不会增加
- ◆ 读操作时，OSD的read速度可达到很高，单个Client因预读窗口的限制没有达到上限，因此，增加Client，读速度线性增长



Ceph评测

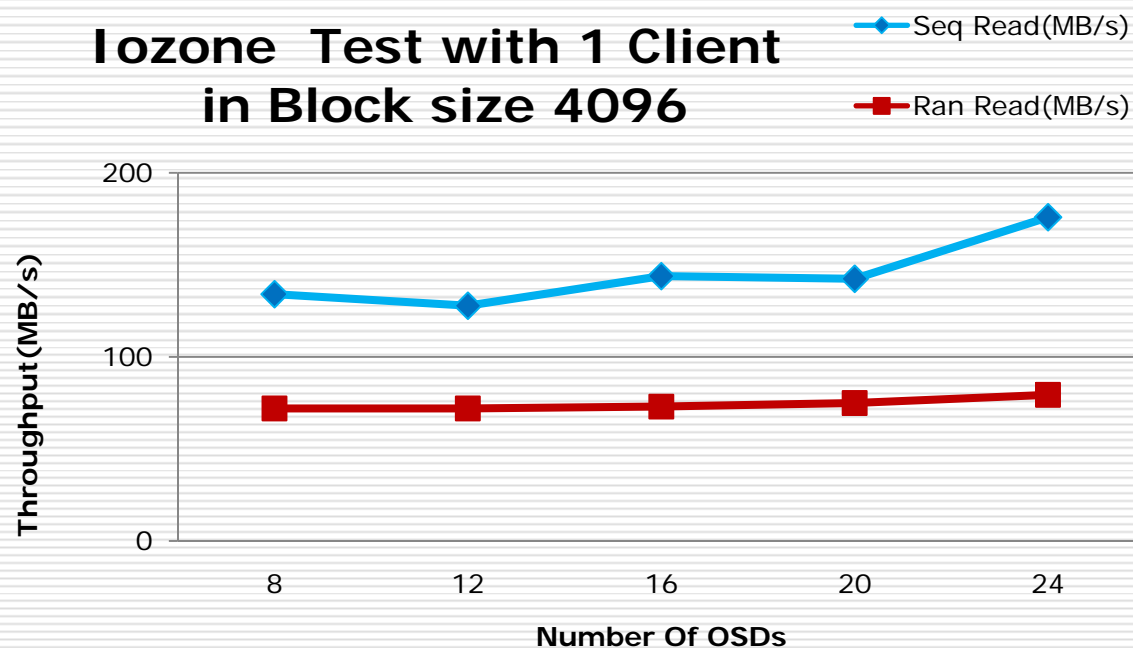
- Max readahead size从8MB增加到256MB





Ceph评测

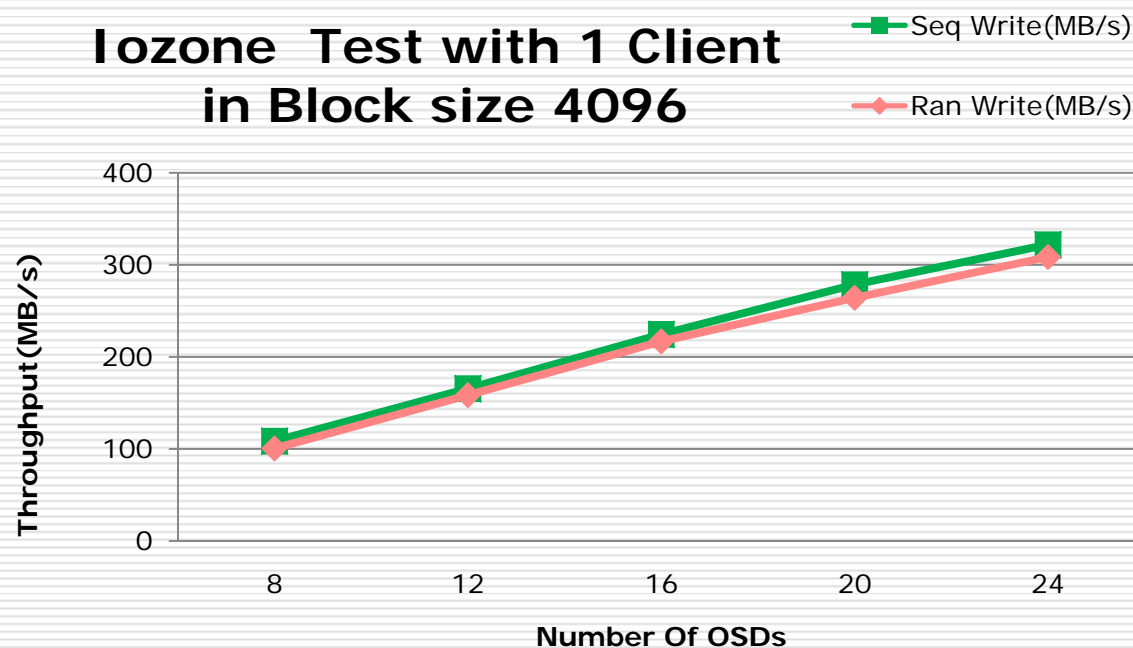
- 固定Client数、Block Size, OSD数增加各项性能变化曲线如下:





Ceph评测

- 固定Client数、Block Size, OSD数增加各项性能变化曲线如下:





评测分析

■ 条件

- ◆ 固定Client个数（1个），固定Block Size（4MB）

■ 结论

- ◆ 随着OSD数量增加，读性能基本平稳，随机读速度几乎不变，顺序读和跳读速度有较小波动
- ◆ 随着OSD数量增加，写速度呈线性增长

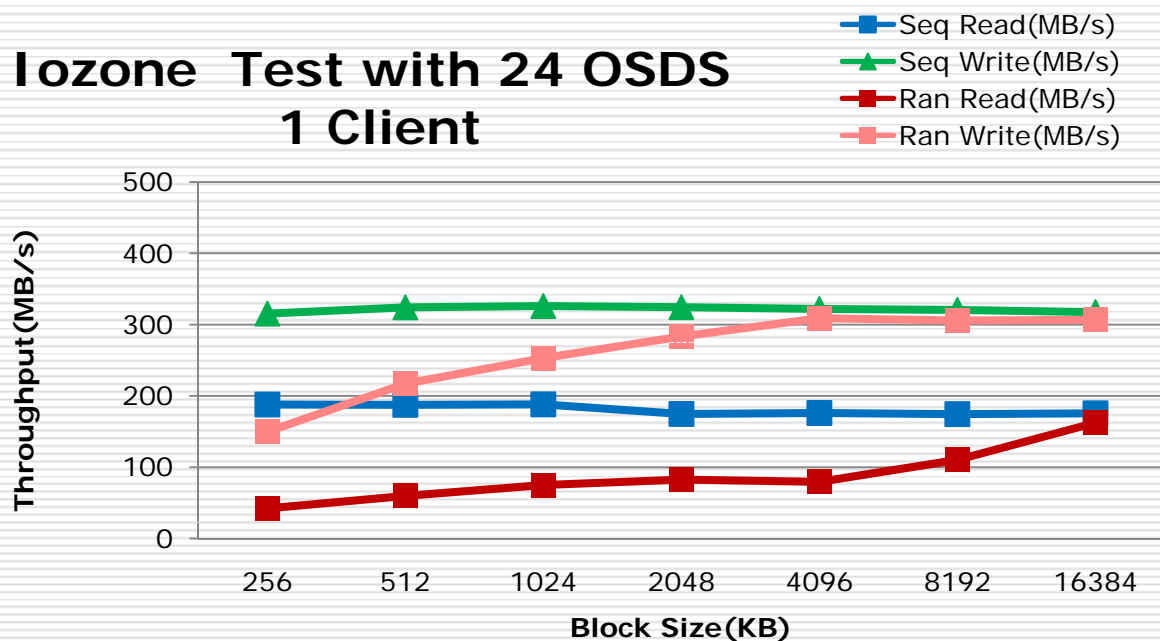
■ 分析

- ◆ 读操作时，受限于预读窗口，增加OSD性能改变不明显
- ◆ 写操作时，更多的OSD能提供更高的写带宽，因此性能增加明显



Ceph评测

- 固定Client数、OSD数，Block Size 增加各项性能变化曲线如下：





评测分析

■ 条件

- ◆ 固定Client个数（1个），固定OSD数（24个）

■ 结论

- ◆ 随着Block Size变大，顺序读写速度改变不大，随机读和随机写速度呈上升曲线变化
- ◆ 随着Block Size变大，跨步读速度不是一直上升，在Block size为16M时出现下降

■ 分析

- ◆ 随机读写随着block size增大，通信次数减少，性能提升
- ◆ 顺序读写受预读和缓存影响，随着block size增大性能改变不明显



Inline data support

■ 测试环境:

◆ 操作系统: Ubuntu 12.04 x86_64

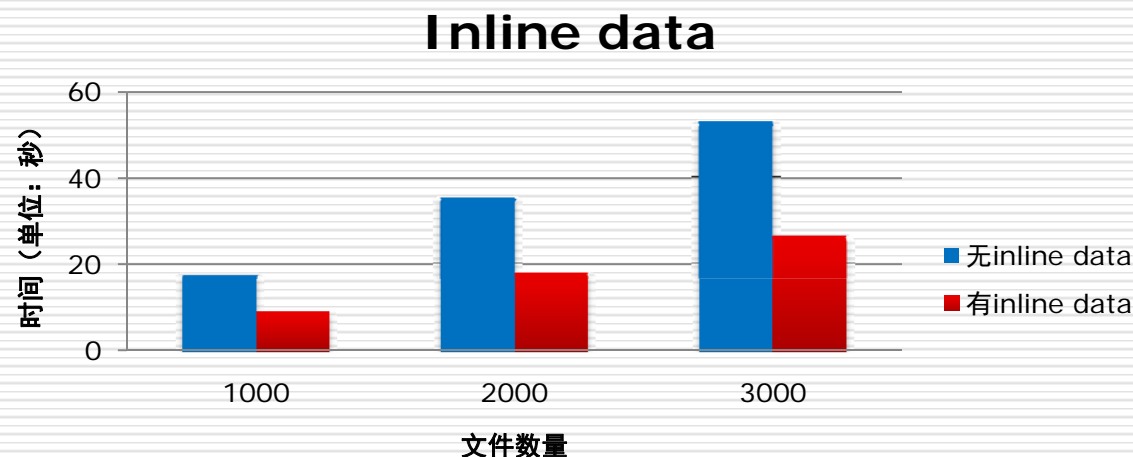
◆ ceph版本: 0.56.6

◆ Ceph部署: 15个OSD, 1个MDS, 1个MON

■ 测试方法:

◆ 顺序读取一定数量的1k小文件, 统计总共花费的时间

■ 测试结果:





谢谢