誉天教育
YU TIAN EDU

Linux云计算架构师涨薪班

Ceph原理及架构

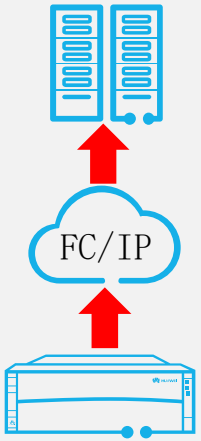
学习目标

- Sever SAN和传统存储对比
- Server SAN的应用场景
- 存储的分类
- ceph的简介与哲学
- ceph的架构及组件
- ceph集群的访问方式
- ceph的文件存储原理
- ceph的组件关键特性

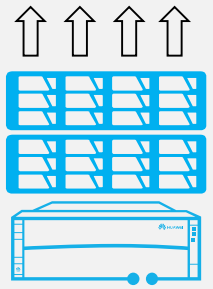
初识Server SAN

- 概念
  - 由多个独立服务器自带的存储组成一个存储资源池，同时融合了计算和存储资源。
- 特征
  - 专有设备变通用设备
  - 计算与存储线性扩展
  - 简单管理、性价比高

传统SAN存储



- 孤立的存储资源：存储通过专用网络连接到有限数量的服务器。



- 存储设备通过添加硬盘框增加容量，控制器性能成为瓶颈。

ServerSAN和传统存储对比

- 从性能对比: Server SAN服务器数量达到一定量可以超越传统存储
- 从稳定性对比：Server SAN 通过软件来维持集群的稳定性
- 从数据可靠性对比：Server SAN基于副本实现高可靠
- 从扩展性对比：Server SAN 扩展性强 轻松达到PB级别
- 从可管理性对比: Server SAN 通常具有web界面 管理便捷
- 从使用场景对比： Server SAN适用于海量存储场景

存储的使用分类

- 块存储: 提供裸磁盘、未被分区格式化使用的磁盘，在主机上的表现形式为一块磁盘
- 文件系统存储: 提供一个可以使用的目录，在主机上的表现形式为共享目录
- 对象存储: 提供一个可以存储文件的接口，通常是http的接口

存储的架构分类

- 集中式存储
  - DAS
  - SAN
  - NAS
- 分布式存储
  - moosefs
  - glusterfs
  - ceph

Ceph起源

- Ceph项目起源于Sage Weil 2003年在加州大学圣克鱼兹分校攻读博士期间的研究课题《Lustre环境中的可扩展问题》
- 2006年，Ceph以LGPLv2协议开源
- 2011年，Inktank公司成立并资助上游开发，并通过其Inktank Ceph Enterprise产品为Ceph提供商业支持
- 2014年，Redhat收购Inktank，Inktank Ceph Enterprise也变为红帽Ceph存储

Ceph简介

- Ceph是提供了软件定义的，统一存储解决方案的开源项目
- Ceph是一个分布式、可扩展、高性能、不存在单点故障的存储系统（支持PB级规模数据）
- 同时支持块存储、文件系统存储、对象存储（兼容swift和S3协议）

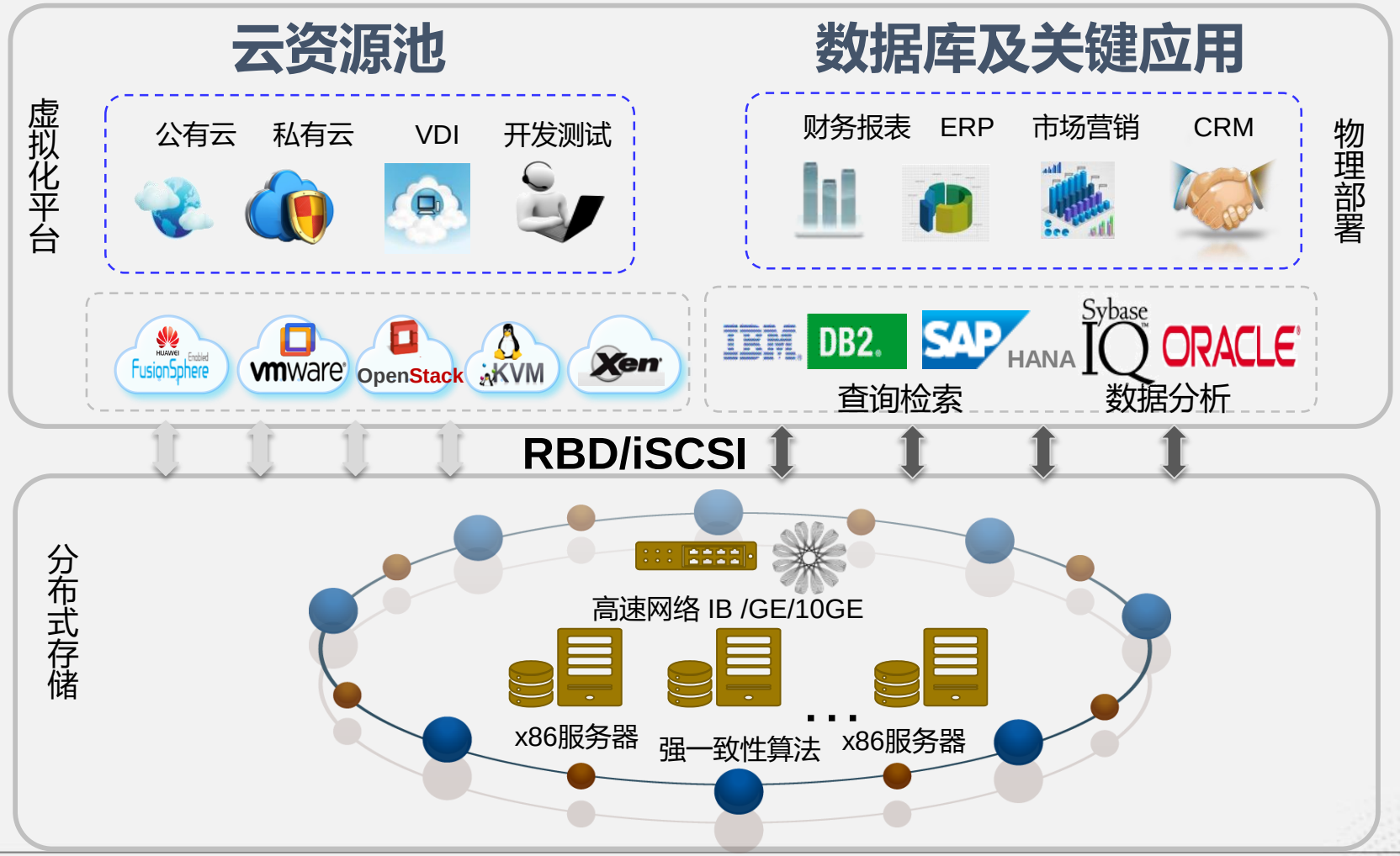
Ceph哲学

- 每个组件必须是可扩展的
- 无任何单点故障（元数据存储服务器可扩展）
- 解决方案必须是基于软件的、开源的、适应性强的
- 运行于现有商业硬件之上
- 每个组件必须尽可能拥有自我管理和自我修复的能力

- 目标：轻松扩展到PB级别、提供不同场景下的高性能存储、数据高可靠性

- 意义： 帮助企业摆脱昂贵的专属硬件

应用场景



Ceph技术创新

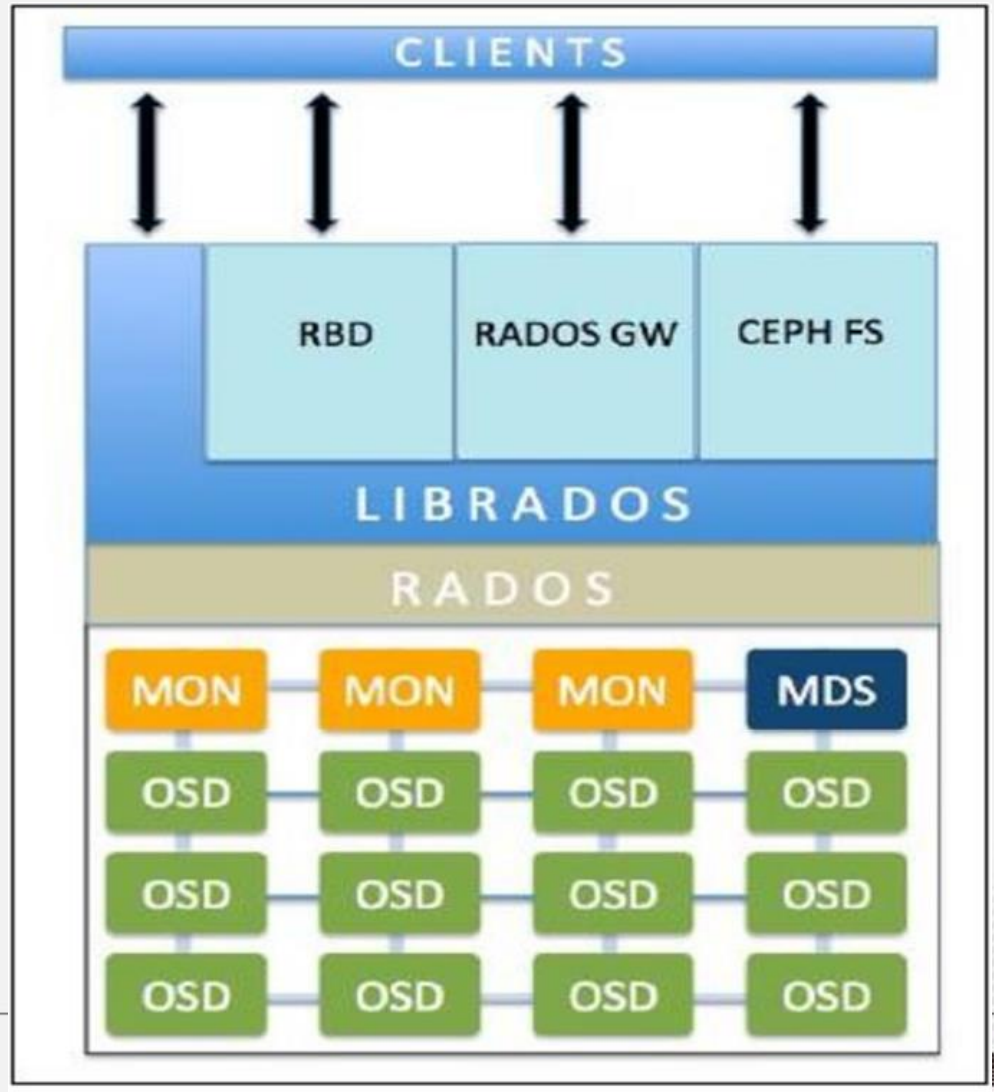
- 采用RADOS（可靠的自主分布式对象存储）系统将数据作为对象存储在逻辑存储池中
- 使用CRUSH（可扩展哈希下的受控复制）算法自动计算应存储对象的位置
- 任何客户端都能够使用CRUSH算法来查找对象的存储位置，因此无需与中央查找服务器通信就能找到对象
- 客户端可以直接与存储对象的Ceph节点通信来获取对象
- CRUSH还能让集群实现自动扩展、数据再平衡、数据保护以及故障恢复

RADOS的一些关键优点

- CRUSH算法能够根据基础架构变化而动态调整
- 复用数据定位快速响应故障
- 无中央查找服务器
- 不需要位置元数据
- 客户端与Ceph节点直接通信
- 多个客户端并行访问，提高吞吐量
- 所有存储设备独立并行运行
- 自动数据保护

Ceph存储后端组件

- Monitors(MON)：维护集群状态的map，帮助其他守护进程互相协调
- Object Storage Devices(OSD)：存储数据，并且处理数据复制、恢复和再平衡
- Managers(MGR)：通过Web浏览器和REST API，跟踪运行指标并显示集群信息，并提供web管理
- Metadata Servers(MDS)：存储供CephFS使用的元数据，能让客户端高效执行POSIX命令



Ceph Monitors

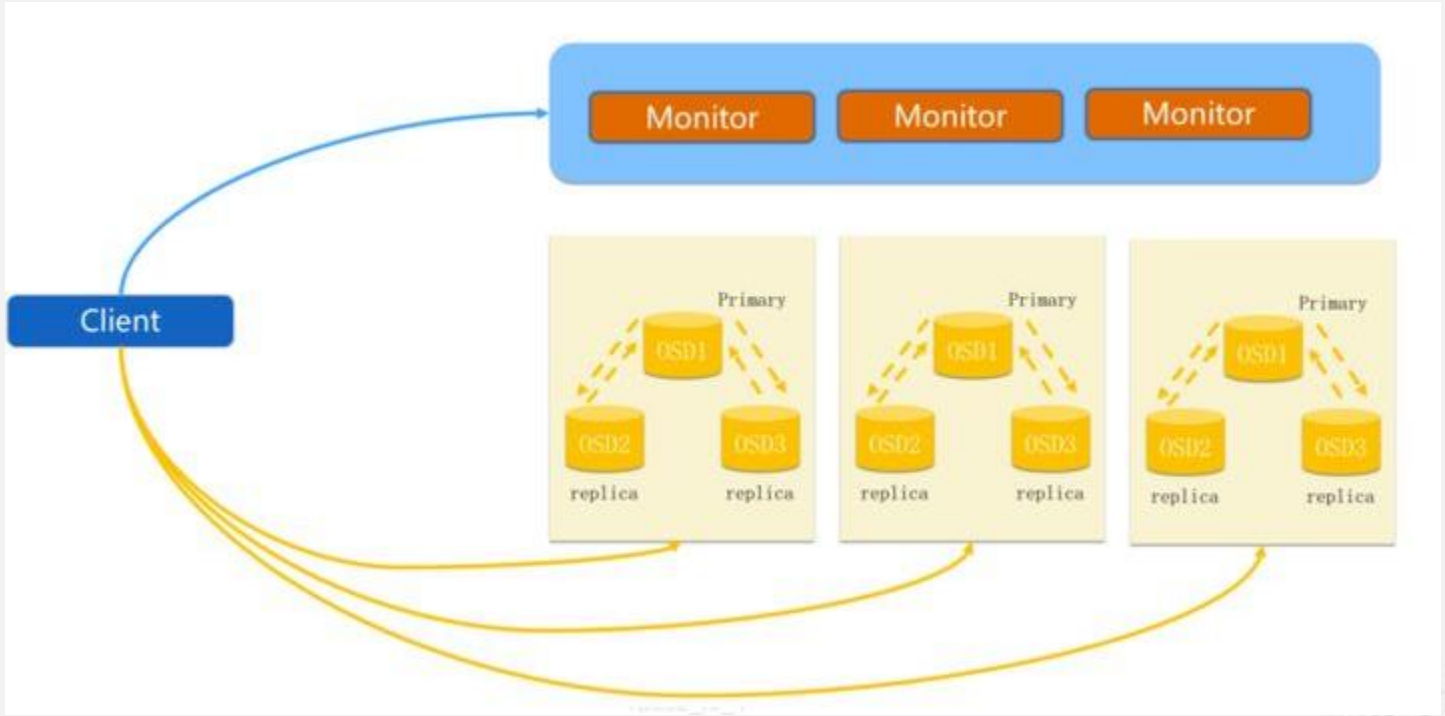
- Ceph monitor通过保存一份集群状态映射来维护整个集群的健康状态。它分别为每个组件维护映射信息，包括OSD map、MON map、PG map和CRUSH map
- 所有集群节点都向MON节点汇报状态信息，并分享它们状态中的任何变化
- Ceph monitor不存储数据
- MON需要配置为奇数个，只有超过半数正常，Ceph存储集群才能运行并可访问

Ceph OSD

- 只要应用程序向Ceph集群发出写操作，数据就会被以对象的形式存储在OSD中。这是Ceph集群中唯一能存储用户数据的组件，同时用户也可以发送读命令来读取数据。
- 通常，一个OSD守护进程会被捆绑到集群中的一块磁盘上。所以在通常情况下，Ceph集群中的物理磁盘的总数，与在磁盘上运行的存储用户数据的OSD守护进程的数量是相同的。
- 每个OSD对应一块磁盘，磁盘会使用文件系统格式化，支持filestore和bluestore驱动。
- OSD设计目标是尽可能接近计算力与物理数据的距离，让集群的性能能够达到最高效。CRUSH算法用于将对象存储到OSD中。
- 对象被自动复制到多个OSD，客户端在读写数据时始终访问primary OSD，其他OSD为secondary OSD，在集群故障时发挥重要作用。

Ceph OSD

- Primary OSD功能:
  - 服务所有I/O请求
  - 复制和保护数据
  - 检查数据的一致性
  - 重平衡数据
  - 恢复数据
- Secondary OSD功能:
  - 行动始终受到Primary OSD的控制
  - 能够变为Primary OSD



Ceph MGR

- MGR提供一系列集群统计数据，在较旧版本的Ceph中，这些数据大部分是由MON收集和维护，负载太高
- 如果集群中没有MGR，不会影响客户端I/O操作，但是将不能查询集群统计数据。建议每集群至少部署两个MGR
- MGR将所有收集的数据集中到一处，并通过tcp的8443端口提供一个web界面对集群进行管理

Ceph MDS

- MDS只为CephFS文件系统跟踪文件的层次结构和存储元数据。MDS不直接提供数据给客户端，从而消除了系统中故障单点。
- MDS使用RADOS存储元数据，MDS本身只在内存中缓存元数据以加速访问
- 访问CephFS的客户端首先向MDS发出请求，以便从正确的OSD获取文件

Ceph访问方式

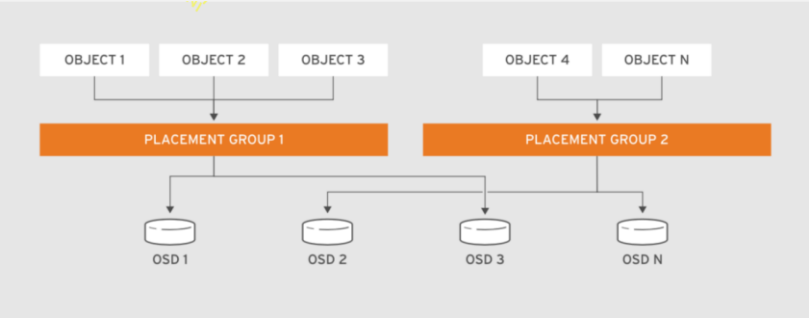
- Ceph原生API(librados)
  - Ceph原生接口，可以通过该接口让应用直接与RADOS协作来访问Ceph集群存储的对象
  - Ceph RBD、Ceph RGW以及CephFS都构建于其上
  - 如果需要最高性能，建议在应用中直接使用librados。如果想简化对Ceph存储的访问，可改用Ceph提供的更高级访问方式，如RGW、RBD、CephFS
- Ceph Object Gateway
  - 利用librados构建的对象存储接口
  - 通过REST API为应用提供网关
  - 支持S3和swift接口
- Ceph Block Device
  - 提供块存储
  - 由分散在集群中的不同的OSD中的个体对象组成
  - Linux内核挂载支持
  - QEMU、KVM和Openstack Cinder的启动支持
- Ceph File System
  - 并行文件系统，元数据由MDS管理

Ceph存储池

- 池是Ceph存储集群的逻辑分区，每个池分配特定数量的hash buckets，将对象分组到一起进行存储，称为PG(Placement Groups)
- 分配给每个池的PG数量可以独立配置，以匹配数据类型以及池所需要的访问权限。PG的数量在创建池的时候配置，可以动态增加，但不能减少。
- CRUSH算法用于选择存放池数据的OSD。每个池分配一条CRUSH规则，用于其放置策略
- 必须为每个请求指定池名称，并且为每个ceph用户授权

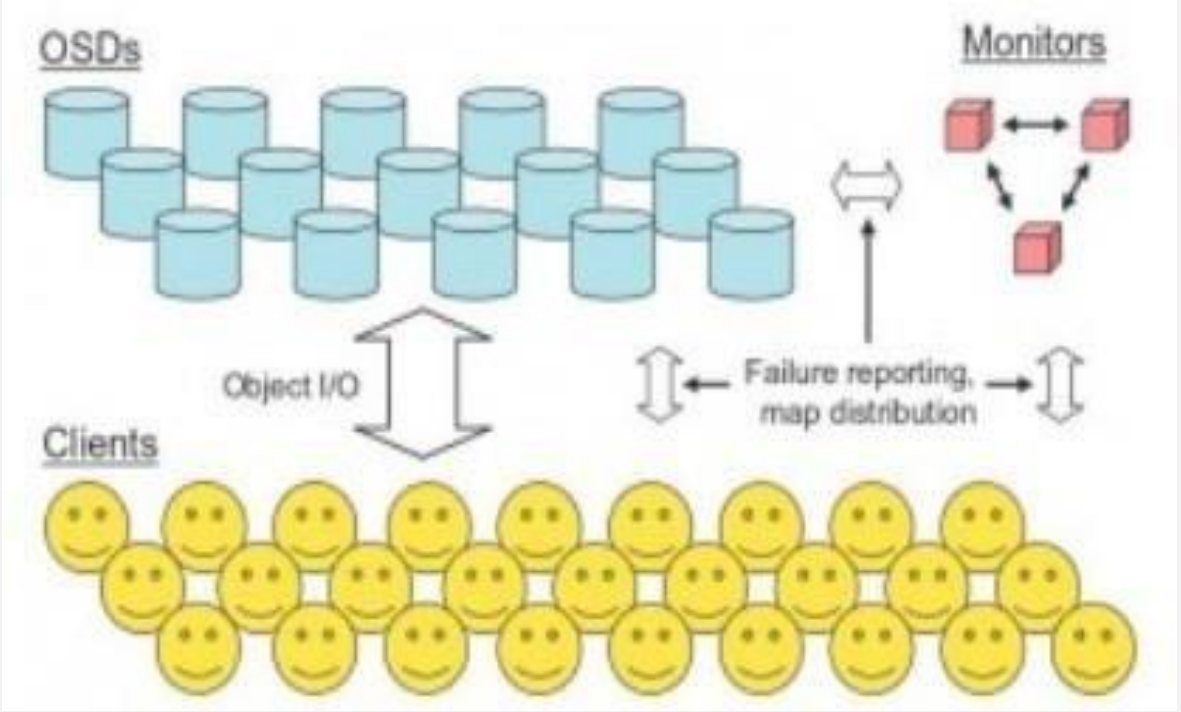
Ceph归置组

- PG将一系列对象聚合到一个hash buckets或组中，并且映射至一组OSD。一个对象仅属于一个PG
- 对象根据对象名称的hash，使用CRUSH算法映射到其PG。这种放置策略称为CRUSH放置规则。放置规则
- 当客户端将对象写入到池时，它使用池的CRUSH放置规则来确定对象的PG。然后客户端使用其cluster map副本、归置组以及CRUSH放置规则来计算对象的副本应写入到哪些OSD中
- 在集群中添加或移除OSD时，PG会自动在正常工作的OSD之间重新平衡



Ceph访问流程

- Ceph客户端从MON获取cluster map副本，这会告知它集群中所有的MON、OSD和MDS
- 获取对象ID以及对象存储池名称，以计算对象存储的PG ID
- 使用CRUSH算法确定哪些OSD负载该PG，并获取primary OSD
- 客户端直接与primary OSD交互来访问对象

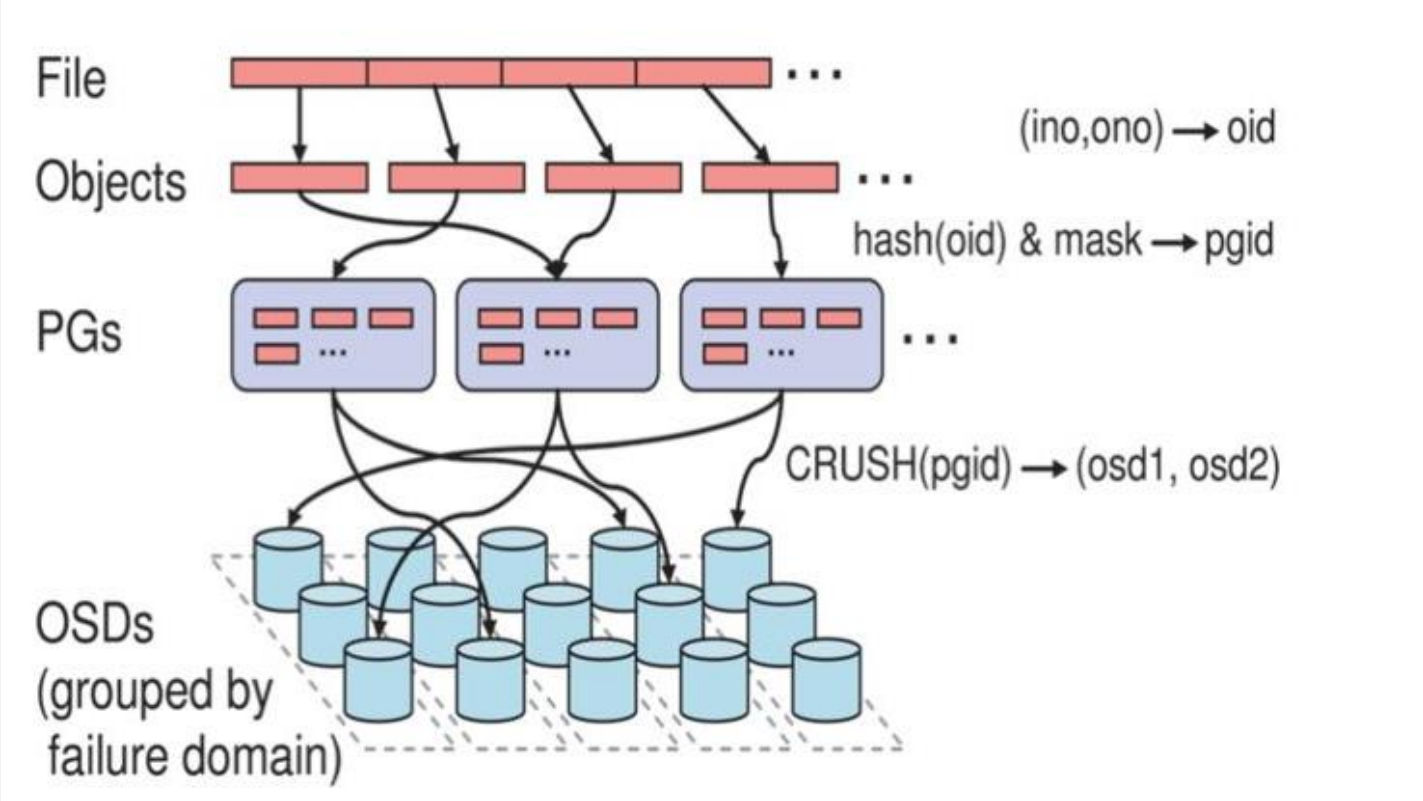


Ceph寻址流程

- File → Object映射：将用户态的file映射成rados能够处理的object，按照统一大小切块并分配id，方便管理以及对多个object并行处理

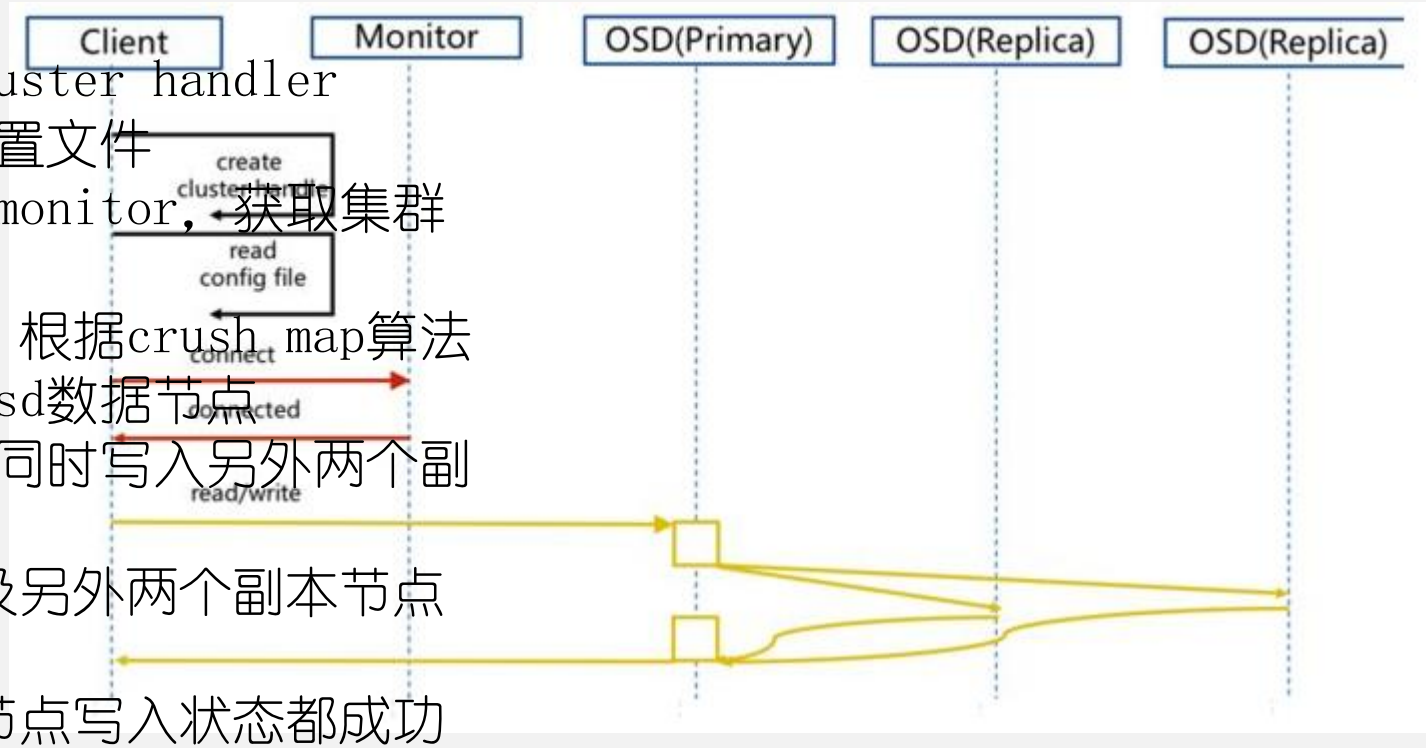
- Object → PG映射：一个PG负责组织若干个object，但一个object只能被映射到一个PG中。同时，一个PG会被映射到n个OSD上，而每个OSD上都会承载大量的PG。 在file被映射为一个或多个object之后，就需要将每个object独立地映射到一个PG中去。当有大量object和大量PG时，RADOS能够保证object和PG之间的近似均匀映射。

- PG → OSD映射：将作为object的逻辑组织单元的PG映射到数据的实际存储单元OSD。RADOS采用CRUSH算法来实现PG到OSD的映射。



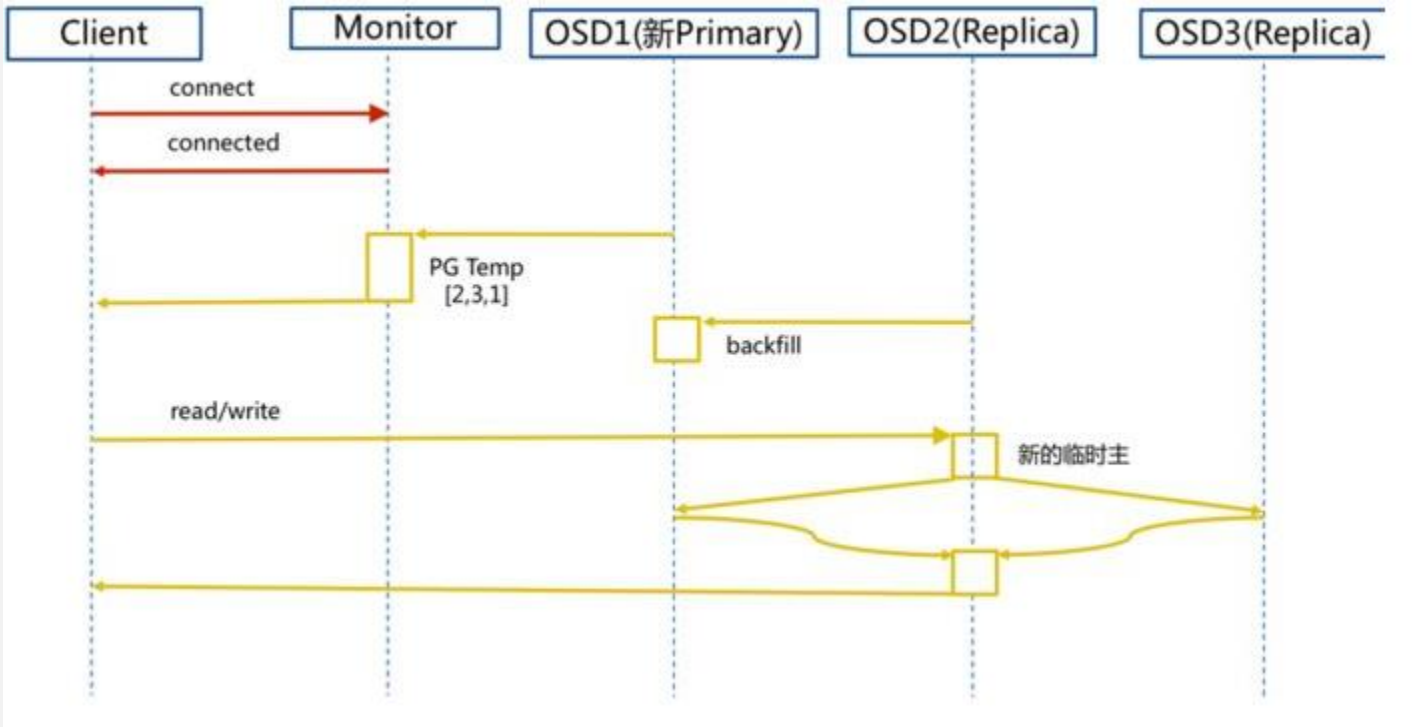
Ceph数据写入流程：正常写入

- client 创建cluster handler
- client 读取配置文件
- client 连接上monitor，获取集群map信息
- client 读写io 根据crush map算法请求对应的主osd数据节点
- 主osd数据节点同时写入另外两个副本节点数据
- 等待主节点以及另外两个副本节点写完数据状态
- 主节点及副本节点写入状态都成功后，返回给client，io写入完成

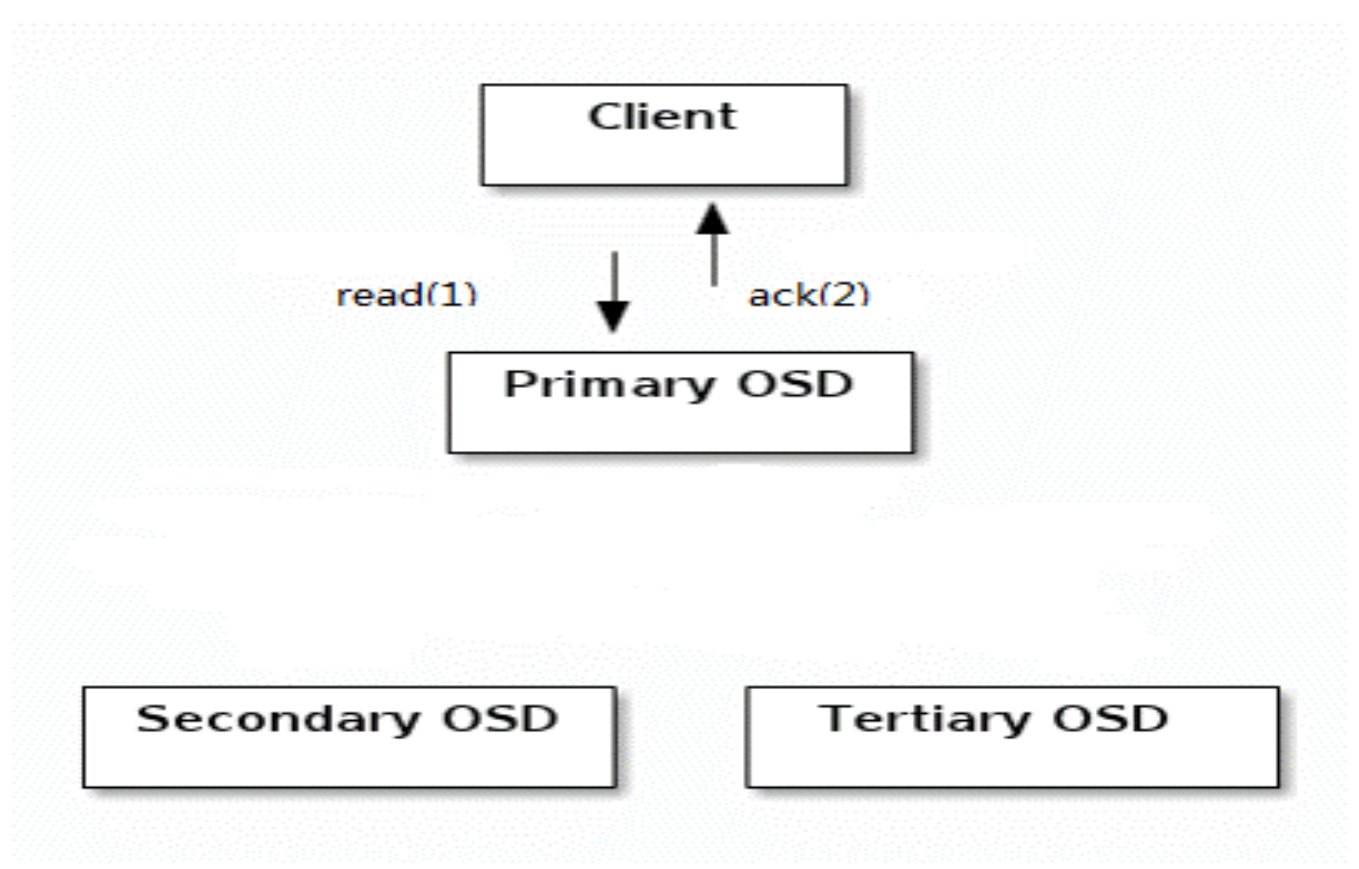


Ceph数据写入流程：主OSD异常

- client连接monitor获取集群map信息
- 同时新主osd1由于没有pg数据会主动上报monitor告知让osd2临时接替为主
- 临时主osd2会把数据全量同步给新主osd1
- client IO读写直接连接临时主osd2进行读写
- osd2收到读写io，同时写入另外两副本节点
- 等待osd2以及另外两副本写入成功
- osd2三份数据都写入成功返回给client，此时client io读写完毕
- 如果osd1数据同步完毕，临时主osd2会交出主角色
- osd1成为主节点，osd2变成副本



Ceph数据读取流程



誉天教育
YU TIAN EDU

Thank you