ceph

Ceph中国社区 | IT大咖说 知识共享平台

CEPHALOCON APAC 2018

THE FUTURE OF STORAGE

22-23 March 2018 | BEIJING

基于Ceph构建EB级对象存储系统

张冬卯

议程

1. radosgw遇到的挑战
2. radosgw的tunning
3. yig的架构设计
4. 性能的挑战

radosgw遇到的挑战

1. 单bucket下面文件数目的限制
2. bucket下面文件过多时， List操作延迟变高
3. 后台ceph集群在做recovery或者backfill时极大影响系统性能
4. 如何提高文件读写性能?
4. 线上多个radosgw能否共享cache?

radosgw 的tunning

```
while (http.read(buf)) {
        write(rados, buf);
}
```

1. async write
2. bufferlist + writev

```
while (http.read(buf)) {
        bufferlist.append(buf);
        if (bufferlist.is_big_enough()) {
                rados.async_write(bufferlist);
        }
}
```

radosgw 的tuning

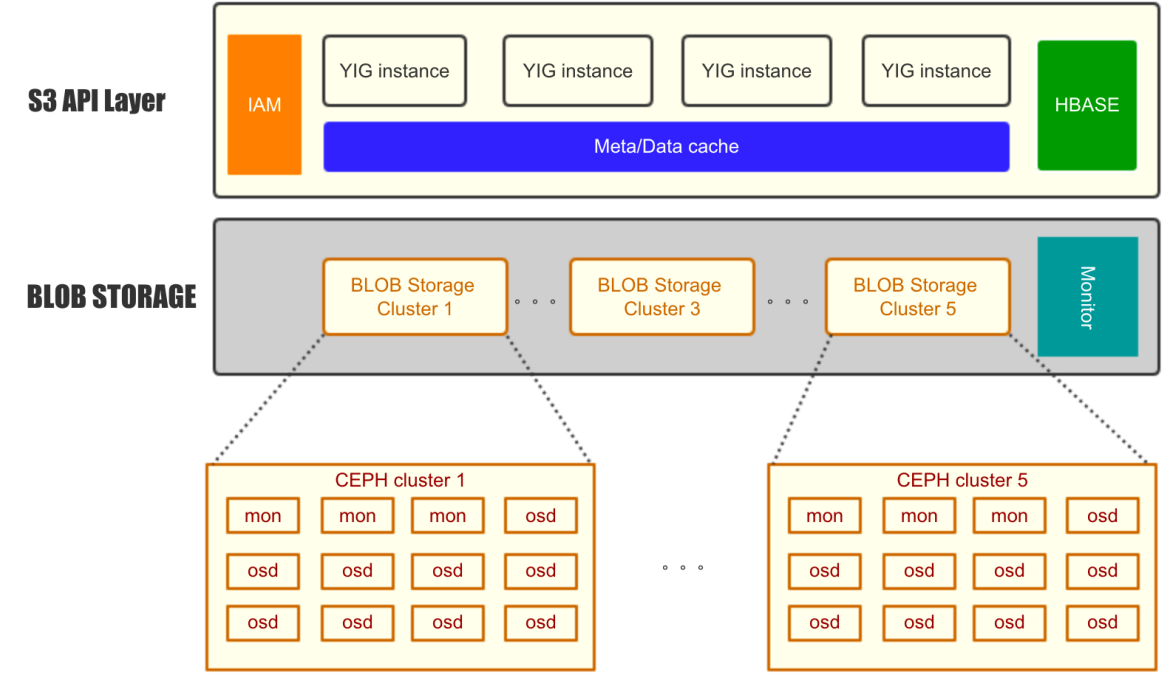
1. 提升.rgw，.rgw.index这些关键元数据pool的性能
2. 辅助功能外移，如日志, 统计容量和用户认证模块
3. 更好的cache

Why yig?

1. radosgw的metadata分布在omap, xattr, object, 难以debug
2. 用C++实现繁杂的S3接口, 代码复杂度越来越大

- Yig同时接入多个ceph集群, 屏蔽ceph rebalance的影响
- Yig连接统一的cache层, 提高响应速度
- Yig统一元数据层到分布式的数据库

Yig 设计


S3 API Layer
IAM
YIG instance
YIG instance
YIG instance
YIG instance
HBASE
Meta/Data cache
BLOB STORAGE
BLOB Storage Cluster 1
BLOB Storage Cluster 3
BLOB Storage Cluster 5
Monitor
CEPH cluster 1
mon
mon
mon
osd
osd
osd
osd
osd
osd
osd
osd
CEPH cluster 5
mon
mon
mon
osd
osd
osd
osd
osd
osd
osd
osd

Ceph提升大文件性能

1. yig使用动态的buf，根据用户上传的速度的大小调整buf，在(512K~1M)之间

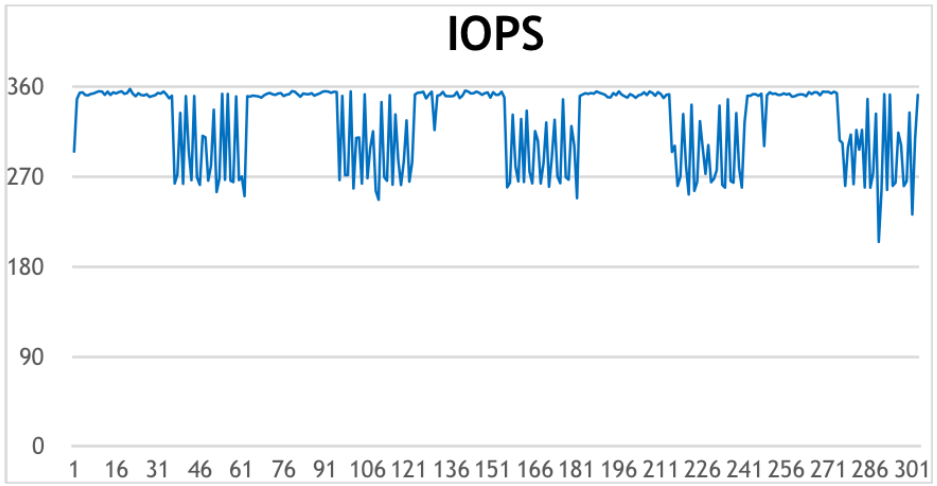
2. 用striperados API提高并发级别

3. EC pool有更快的写性能，但是更慢的恢复时间

ceph提升小文件性能

1. 用kvstore/bluestore，底层用rocksdb + bluefs
2. PATCH: rados增加新FLAG:　LIBRADOS_OP_FLAG_FADVISE_NEWOBJ, ceph的写入之前, 不再尝试读取文件, CPU占用降低了50%
3. 增加rocksdb的sst文件大小

ceph提升小文件性能



纵轴为IOPS　横轴为时间轴，单位为10s

1. 3台服务器, 普通SATA硬盘
2. 文件大小1K
3. 10% write, 90% read

Netropy：高性能 Datastore

1. EC不适合小文件, 所以仍旧采用多副本的方式
2. ceph强一致模型, 小文件写入的延迟偏大, 采用raft protocol 写入2副本就返回, 提高响应时间
3. 目前的底层存储引擎rocksdb, lmdb等都不太适合1K ~ 256K的value存储
4. ceph recovery方式是按照key遍历,每个key依次拷贝, 在小文件的情况下, 速度很慢.

FAST2016: WiscKey: Separating Keys from Values in SSD-conscious Storage

ceph

Ceph中国社区 | IT大咖说 知识共享平台

CEPHALOCON APAC 2018

THE FUTURE OF STORAGE

22-23 March 2018 | BEIJING

Thank You

张冬卯