**CephFS架构解读与测试分析(下）**

原创 *2017-09-27* *杨冠军* [日知录技术社区](https://mp.weixin.qq.com/s?__biz=MzI2NjU4MzkwNA==&mid=2247483975&idx=1&sn=64bfd93514f7961754ce9b43af26238b&chksm=ea8aa848ddfd215e536d2b23beaf4607da11e3810837c8e6afc3b47d70ea9a72734c6c653ffd&mpshare=1&scene=23&srcid=0927OvaNaUgXsAN8hvxXNkb9##)





    本文专业性较强，略枯燥，故配美图两张调节一下。

**【编者按】**日知录技术社区于8月23日举行了线上分享活动，由资深存储人士杨冠军作了题为**“CephFS架构解读与测试分析”**的分享。本次分享受到了存储行业很多人士的关注，冠军深厚的技术背景和渊博的知识面受到了大家的一致好评。经沟通，由冠军将本次分享的内容整理成文章发布在日知录技术社区微信公众号上。由于文章较长，我们分两次刊登。本篇为下半部分，讲述CephFS测试相关问题，包括测试目的、测试方法、测试工具、测试结果及分析。最后是总结与展望。

**【作者介绍】**杨冠军，09年硕士毕业于中科院计算所网络存储专业，专注于存储领域十多年，曾就职于中科院计算所存储研究中心、华赛、EMC、蓝汛、云英等公司，先后从事过NAS、SAN、分布式存储系统的研发，对存储系统的技术、体系架构、应用有深刻的理解。主要专注于Ceph和Openstack相关的公有云、私有云技术领域。

**CephFS测试**

      为了了验证CephFS是否满足产品需求，我们基于最新的Ceph Jewel 10.2.7版本做了了测试。

**1、CephFS Jewel版本特性**

**1）**CephFS – Production Ready

**2）** Single Active MDS，Active-Standby MDSs

**3）** Single CephFS within a single Ceph Cluster

**4）** CephFS requires at least kernel 3.10.x

**5）**Experimental Features

      Multi Active MDSs

      Multiple CephFS file systems within a

             single Ceph Cluster

      Directory Fragmentation

**2、CephFS测试目的**

**1）**  CephFS POSIX基本功能完备？

**2）**  CephFS性能跑满整个集群？

**3）**  CephFS长时间是否稳定？

**4）**  CephFS能否应对MDS异常？

**NO:**

 不是针对MDS的参数调优

 不是MDS的压力测试

     MDS压力测试时建议配置在单独的机器器上

     调大 mds\_cache\_size

 MDS压力测试请参考

        https://www.slideshare.net/XiaoxiChen3/cephfs-jewel-mds-performance-benchmark

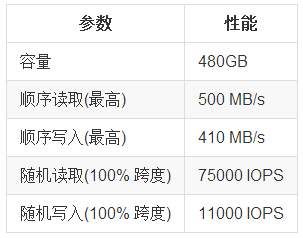
**3、CephFS测试环境**

针对测试目的，我们选择了了三台物理理机搭建一个全新的Ceph集群，提供CephFS服务。

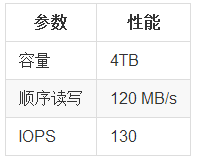
**3.1**物理机的配置

* 10个4T 7200RPM SATA盘
* 2个480GB的SATA SSD盘，Intel S3500
* 2个万兆网卡

  SSD盘的性能为：

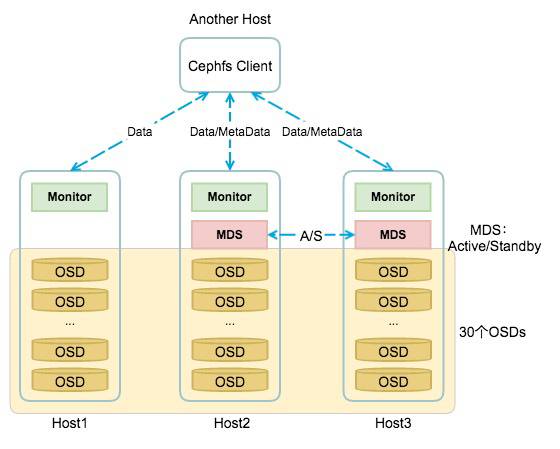


 SATA盘的性能为：



**3.2 Ceph配置参数**

测试用的CephFS client为单独的服务器，128G内存，万兆网络连接Ceph集群



如上图所示，Ceph集群的部署配置为：

* replica为3
* 三个Monitor
* 两个MDS部署为Active/Standy

| **模块** | **版本** |
| --- | --- |
| Ceph Version | Jewel 10.2.7 |
| Ceph Cluster OS | CentOS Linux release 7.2.1511 (Core) |
| Ceph Cluster kernel version | 3.10.0-327.el7.x86\_64 |
| Cephfs Client OS | CentOS Linux release 7.2.1511 (Core) |
| Cephfs kernel version | 4.11.3-1.el7.elrepo.x86\_64 |

Ceph集群和CephFS client的系统版本信息如下：

**3.3 预估Ceph集群性能**

通过Ceph集群架构和物理磁盘、网络的性能指标，就可以预估整个Ceph集群的性能了。

如我们这个Ceph集群，三台物理机，配置三副本，又受限于单Client端的万兆网络性能，所以整个集群的最大吞吐量为：单台物理机上的磁盘性能 / 万兆网络性能的最小值。

每个物理机上，2个SSD做10个OSD的journal，其整体性能约为：  
  
2 \* (单个ssd盘的性能) / 10 \* (单个sata盘的性能) / 万兆网络性能 的最小值。

**4、CephFS测试工具**

1. 功能测试

手动，fstest

1. 性能测试

dd，fio，iozone，filebench

1. 稳定性测试

fio，iozone，自写脚本

1. 异常测试

手动

**5、CephFS测试分析**

**5.1 功能测试**

1. 手动

我们使用文件系统的常用操作：mkdir/cd/touch/echo/cat/chmod/chown/mv/ln/rm等。

1. fstest

fstest是一套简化版的文件系统posix兼容性测试条件，有3600来个回归测试，测试的系统调用覆盖的也比较全面。  
chmod, chown, link, mkdir, mkfifo, open, rename, rmdir, symlink, truncate, unlink

**结论**  
  
功能测试通过

**5.2 性能测试**

性能测试比较重要，也是我们测试的重点，按照CephFS的Layout配置，我们选择了三类Layout配置：

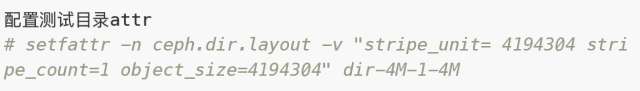
1）stripe\_unit=1M, stripe\_count=4, object\_size=4M

    目录为: dir-1M-4-4M，条带大小为1M，条带数目为4，object大小为4M



2） stripe\_unit=4M, stripe\_count=1, object\_size=4M

   目录为: dir-4M-1-4M，也是系统默认配置



3）stripe\_unit=4M, stripe\_count=4, object\_size=64M

   目录为: dir-4M-4-64M，条带大小为4M，条带数目为4，object大小为64M



注：后续图表中分别拿上述目录名来代表三种CephFS Layout配置分类

* dir-1M-4-4M
* dir-4M-1-4M
* dir-4M-4-64M

**dd**

linux系统常用的测试设备和系统性能的工具。

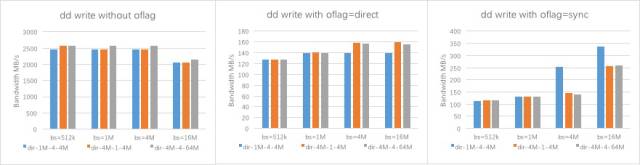
**测试命令：**

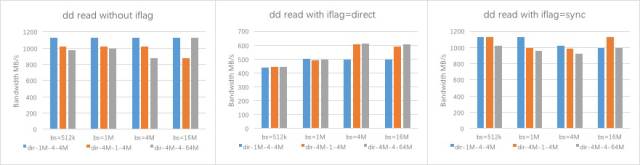
* Direct IO： oflag/iflag=direct
* Sync IO：oflag/iflag=sync
* Normal IO：不指定oflag/iflag

测试文件大小：20G

不能选择太小的测试文件，减少系统缓存的影响

**测试结果：**





**结论：**

1. normal io，客户端缓存影响，写性能较高，不做分析。读性能约为1GB/s，受限于万兆网卡。
2. direct io，客户端写性能只有150MB/s，读性能只有600MB/s，这个是cephfs kernel client端的direct io逻辑导致的。
3. sync io，随着bsd的增大性能有所提升，写性能能到 550MB/s，读性能有1GB/s
4. Stripe模式变化的角度分析
   * bs=512k/1M时，各个stripe模式下的IO性能基本相同
   * bs=4M/16M时，针对direct io，stripe unit=1M的条带性能略低（kernel client的direct io逻辑有关），针对sync io，stripe unit=1M的条带性能较好（并发性较好的原因）
   * 默认的file layout(橙色)，dd的性能就挺好，64Mobjcet 的stripe模式(灰色)没有明显的性能提升

**fio**

fio也是我们性能测试中常用的一个工具，详细介绍Google之。

我们测试中固定配置：

-filename=tstfile   指定测试文件的name

-size=20G           指定测试文件的size为20G

-direct=1           指定测试IO为DIRECT IO

-thread             指定使用thread模式

-name=fio-tst-name  指定job name

测试bandwidth时：

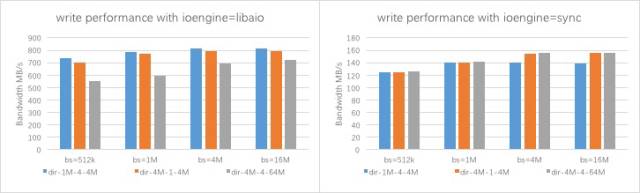
1. -ioengine=libaio/sync
2. -bs=512k/1M/4M/16M
3. -rw=write/read
4. -iodepth=64 -iodepth\_batch=8 -iodepth\_batch\_complete=8

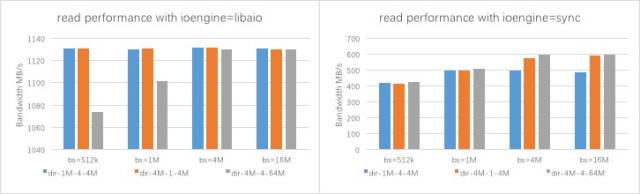
测试iops时：

1. -ioengine=libaio
2. -bs=4k
3. -runtime=300
4. -rw=randwrite/randread
5. -iodepth=64 -iodepth\_batch=1 -iodepth\_batch\_complete=1

**测试结果：**

bandwidth的测试结果如下图：





* bandwidth: direct sync IO
  + write/randwrite 性能最多约为：155 MB/s
  + read/randread 性能最多约为：600 MB/s
* bandwidth: direct libaio

这基本就是集群的整体性能

* + write/randwrite 性能最多约为：810 MB/s
  + read/randread 性能最多约为：1130 MB/s
* bandwidth: cephfs stripe模式变化时

结论与dd的基本相同

iops的测试结果如下表：

| **io mode** | **type** | **dir-1M-4-4M** | **dir-4M-1-4M** | **dir-4M-4-64M** |
| --- | --- | --- | --- | --- |
| randwrite | iops | 4791 | 4172 | 4130 |
|  | latency(ms) | 13.35 | 15.33 | 15.49 |
| randread | iops | 2436 | 2418 | 2261 |
|  | latency(ms) | 26.26 | 26.46 | 28.30 |

注释：上诉测试randread中，因为有cephfs这一层，所以即使direct IO，在OSD上也不一定会read磁盘，因为OSD有缓存数据；所以这里测试采取每次测试前在所有ceph cluster的host上执行sync; echo 3 > /proc/sys/vm/drop\_caches;清理缓存；

**iozone**

iozone是目前应用非常广泛的文件系统测试标准工具，它能够产生并测量各种的操作性能，包括read, write, re-read, re-write, read backwards, read strided, fread, fwrite, random read, pread ,mmap, aio\_read, aio\_write等操作。

1. 测试DIRET IO / SYNC IO - 非throughput模式

不指定threads，测试单个线程的iozone性能

iozone -a -i 0 -i 1 -i 2 -n 1m -g 10G -y 128k -q 16m -I -Rb iozone-directio-output.xls

iozone -a -i 0 -i 1 -i 2 -n 1m -g 10G -y 128k -q 16m -o -Rb iozone-syncio-output.xls

1. 测试系统吞吐量 - throughput模式

指定threads=16，获取整个系统的throughput

iozone -a -i 0 -i 1 -i 2 -r 16m -s 2G -I -t 16 -Rb iozone-directio-throughput-output.xls

iozone -a -i 0 -i 1 -i 2 -r 16m -s 2G -o -t 16 -Rb iozone-syncio-throughput-output.xls

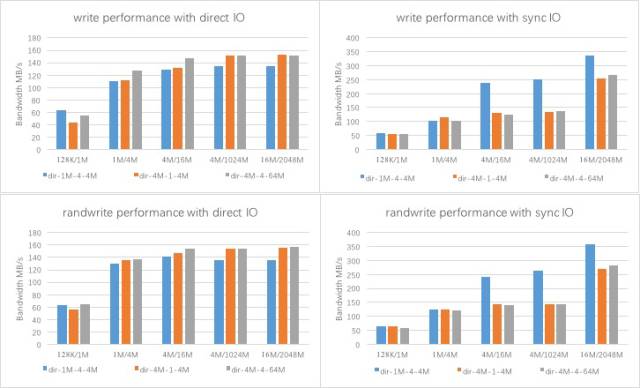
**测试结果：**

iozone测试的结果很多，很难每个都画出图表展示出来，这里挑选几组对比数据作为对比。

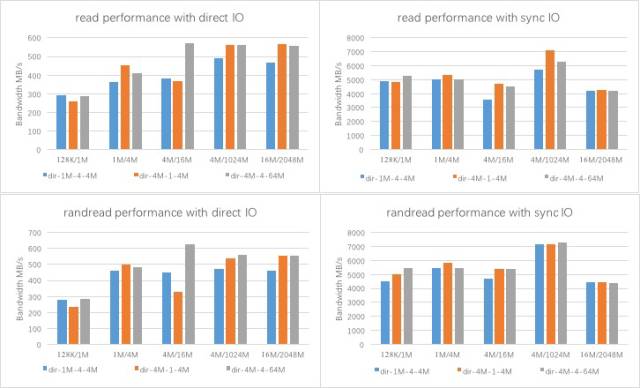
下图中的类似128K/1M的文字含义为：记录块为128K，测试文件为1M  
  
性能输出的单位为：MB/s

* 非throughput模式

       write性能



read性能

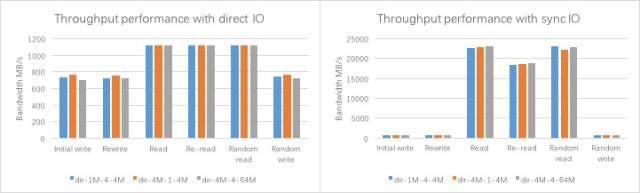


1. 写性能：direct IO模式为 150 MB/s，sync IO模式为 350MB/s
2. 读性能：direct IO模式为 560 MB/s，sync IO模式为 7000 MB/s

（iozone的io模式和client端缓存的影响，指标不准确）

1. Stripe模式变化： 各个Stripe下性能基本一致，对于小文件小IO模式来说，dir-1M-4-4M的性能略好

* throughput模式

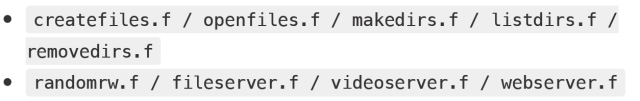


1. 各种write的性能基本相同，最大约为 750 MB/s，基本是集群写的极限
2. direct IO模式下，读性能约为 1120 MB/s，client端万兆网络带宽的极限
3. sync IO模式下，读性能高达 22500 MB/s，iozone的io模式和client端缓存的影响，指标不准确

**filebench**

filebench是一款文件系统性能的自动化测试工具，它通过快速模拟真实应用服务器的负载来测试文件系统的性能。

filebench有很多定义好的workload，针对cephfs的测试，我们可以选择其中一部分有代表性的workloads即可。



**结论**

1. filebench测试用例，除了读写操作外，其他的都是元数据操作，基本不受cephfs stripe的影响
2. 各种文件操作的时延都不高，可以满足基本的对filesystem的需求

**5.3  稳定性测试**

为了测试cephfs是否能在线上提供服务，我们需要测试下其稳定性，这里采用两种方式测试。

**读写数据模式**

针对读写数据模式，我们选择工具fio，在cephfs client端长时间运行，看会不会报错。

测试逻辑大概如下：

# fio循环测试读写

while now < time

   fio write 10G file

   fio read 10G file

   delete file

**读写元数据模式**

针对读写元数据模式，我们采用自写脚本，大规模创建目录、文件、写很小数据到文件中，在cephfs client端长时间运行，看会不会报错。

测试逻辑大概如下：

# 百万级别的文件个数

while now < time

   create dirs

   touch files

   write little data to each file

   delete files

   delete dirs

**结论**

* 通过几天的连续测试，cephfs一切正常，这说明cephfs是可以应用到生产环境的。
* 至于上亿级别的文件测试，也遇到点问题。

**问题与解决**

* 日志中报Behind on trimming告警  
    
  调整参数

    mds\_log\_max\_expiring，

    mds\_log\_max\_segments

* rm删除上亿文件时报No space left on device错误  
    
  调大参数

    mds\_bal\_fragment\_size\_max，

    mds\_max\_purge\_files，mds\_max\_purge\_ops\_per\_pg

* 日志中报\_send skipping beacon, heartbeat map not healthy  
    
  调大参数

    mds\_beacon\_grace，mds\_session\_timeout，

    mds\_reconnect\_timeout

**基本思路：**

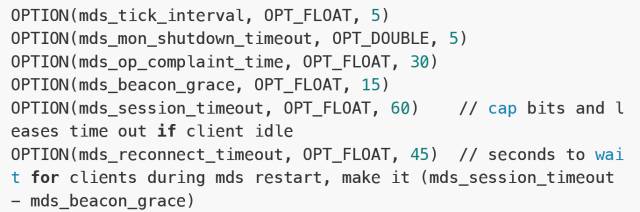
1. 查看Client和MDS端log
2. Google搜索关键字 or 搜索Ceph相关代码
3. 分析原因
4. 调整参数

当然也有的问题不是简单调整参数就搞定的，那就尽量去分析问题，向社区提bug反馈。

**5.4 异常测试**

cephfs的功能依赖于MDS和Ceph Cluster，关键的元数据都通过MDS获取，这里测试的异常也主要基于MDS的异常进行分类的。

查看ceph MDS与interl和timeout相关的配置有：



在Sage weil的博士论文里提到CephFS允许客户端缓存metadata 30s，所以这里测试对MDS stop/start的时间间隔取为：2s，10s，60s

测试工具：fio

**测试分类：**

1. 单MDS
2. 主从MDS

**测试结果：**

1. 单MDS时：

   2s/10s 无影响

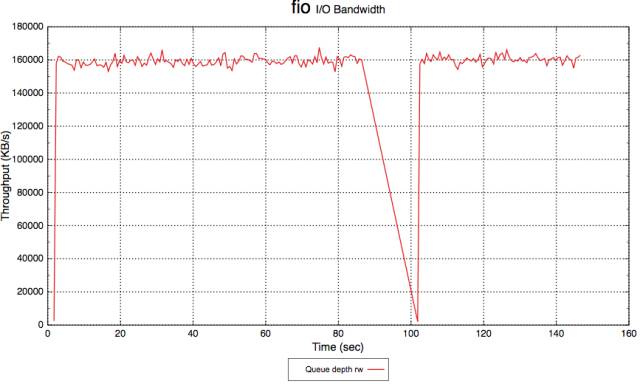
   60s时影响IO

1. 主从MDS时：

  主从不同时停无影响

   同时停时与单MDS一致

mds停60s时会影响IO，fio测试结果如下图：



另外这里只举例说明了fio，同样异常测试中我们也测试了iozone，因为iozone会读写不同的文件，所以在mds停掉后，新的文件操作就会被hang住。

**结论**

1. 单MDS的情况下，短暂的MDS crush并不会影响客户端对一个file的读写
2. 单MDS的情况下，MDS crush后，client端对没有缓存过caps的文件操作会hang住
3. 主从MDS的情况下，只要有一个MDS正常，CephFS的服务就不会中断
4. 主从MDS的情况下，两个MDS都crush后，影响与单MDS的一致

       所以生产环境中，我们建议配置主从MDS的模式，提高CephFS的可用性。

**总结与展望**

**总结**

1. CephFS是production ready的，能满足基本生产环境对文件存储的需求
2. CephFS kernel client端的Linux kernel版本最好大于4.5-rc1（支持aio）
3. 对性能要求不高时，考虑使用CephFS FUSE client，支持Quotas
4. CephFS的主从MDS是稳定的，优于单MDS配置
5. 生成环境使用CephFS时，独立机器上配置MDS，调大“mds\_cache\_size”
6. 使用CephFS时，避免单个目录下包含超级多文件（more than millions）
7. CephFS能跑满整个ceph集群的性能
8. 默认stripe模式下(stripe unit=4M, stripe count=1, object size=4M)， CephFS的性能就挺好
9. 小文件的应用场景下，尝试配置小的stripe unit，对比默认stripe的性能
10. CephFS的Direct IO性能有限，分析后是cephfs kernel client的IO处理逻辑限制的http://www.yangguanjun.com/2017/06/26/cephfs-dd-direct-io-tst-analysis/
11. 受到CephFS client端的系统缓存影响，非Direct IO的读写性能都会比较高，这个不具有太大参考意
12. 使用CephFS kernel client，且object size大于16M时，一次性读取大于16M的数据读时IO会hang住http://www.yangguanjun.com/2017/07/18/cephfs-io-hang-analysis/

**展望**

Ceph Luminous (v12.2.0) - next long-term stable release series

1. The new BlueStore backend for ceph-osd is now stable and the new default for newly created OSDs
2. Multiple active MDS daemons is now considered stable
3. CephFS directory fragmentation is now stable and enabled by default
4. Directory subtrees can be explicitly pinned to specific MDS daemons

**—— End ——**