glusterfs运维经验谈

QQ群：365534424

分布式存储 **Why glusterfs**

- 容量　　PB级
- IO性能好，在我们使用的服务器上能使4个1G的网卡满载，大文件IO性能更好
- 优化参数多，可以根据自己应用和服务器的配置调整优化参数
- 支持多种模式，可以根据应用选择
- 挂载简单灵活
- 文档齐全，社区活跃

glusterfs 基本概念

- volume

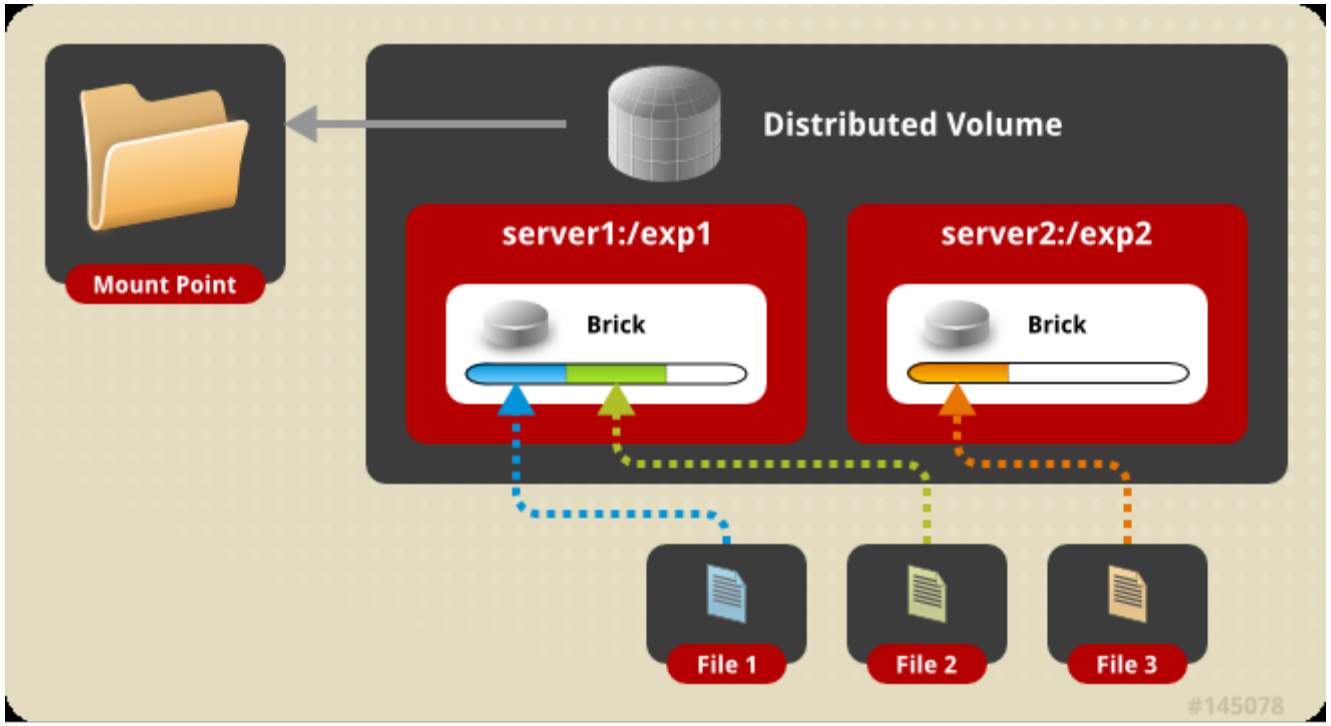
  glusterfs逻辑卷，外部使用者看到的总存储

- brick

  逻辑卷内部的各个物理存储单元，是一个挂载的目录，各个brick以不同的模式组成一个volume

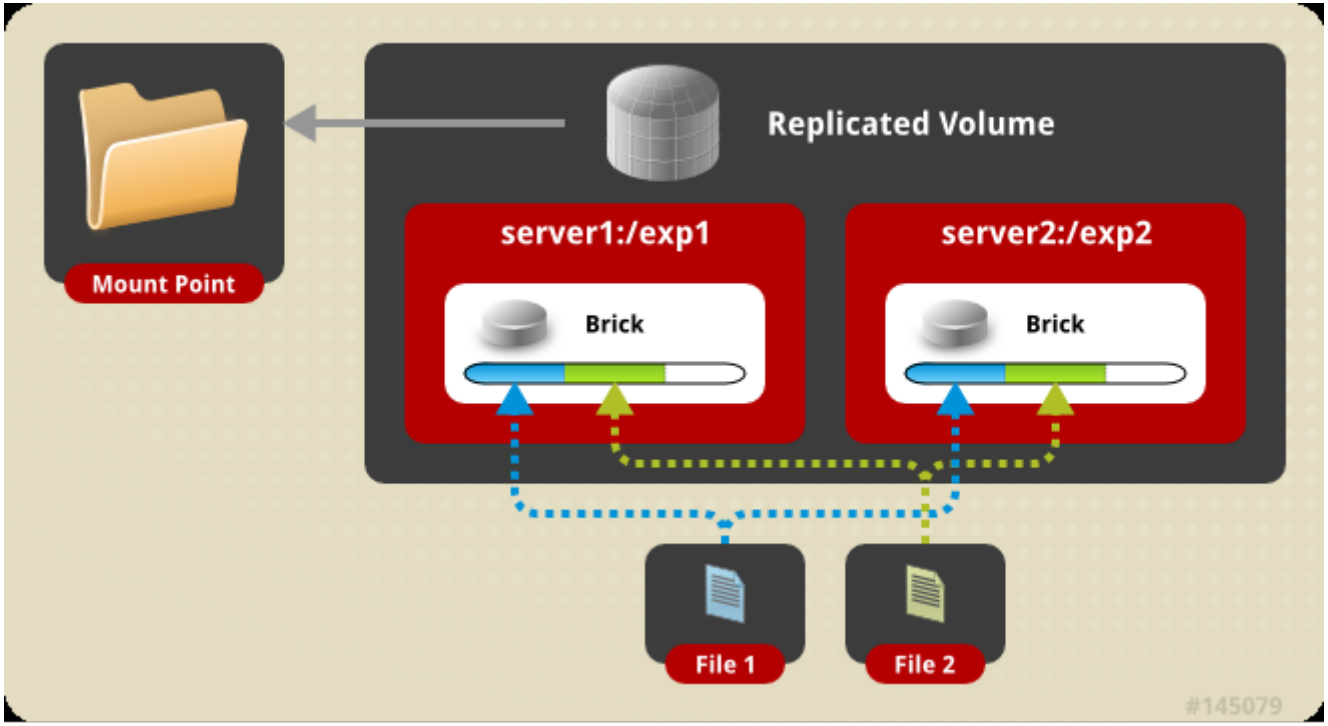
volume 类型 - 简单分布式

- 简单的容量扩展
- 文件被随机放在某一个brick上
- 没有备份



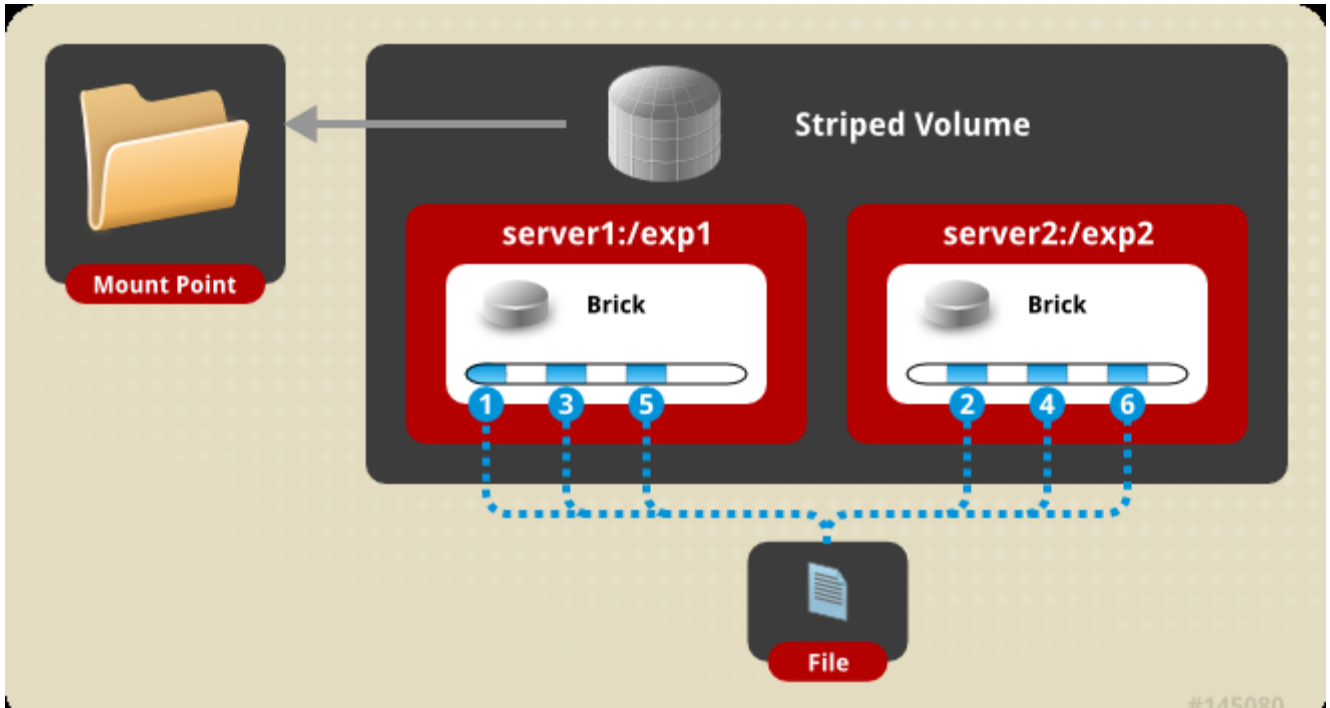
volume 类型 - 简单复制

- 文件被同时放在几个brick上进行备份
- 创建volume的时候可以选择复制的数量
- 减少了容量但提供了高可用的存储



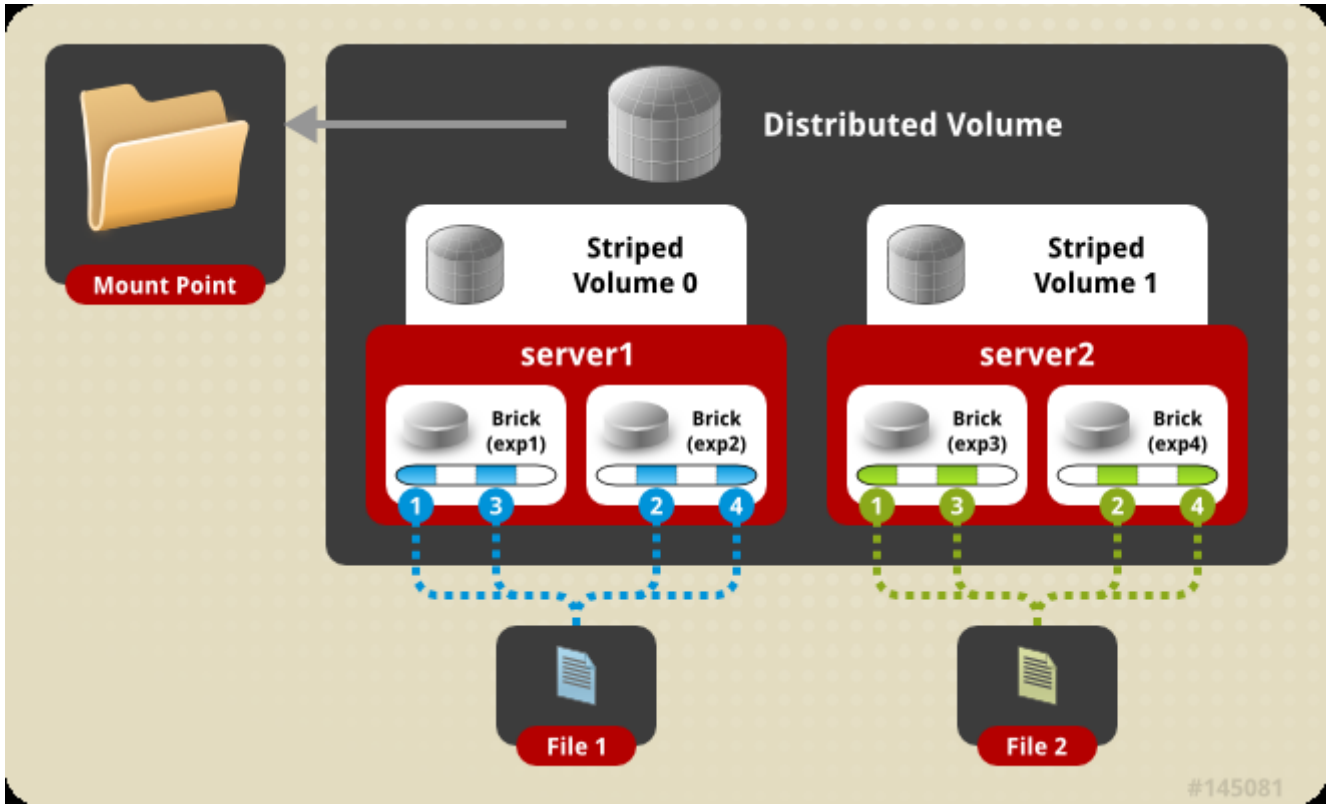
volume 类型 - 条带式

- 文件被分成几份块被存在不同的brick上
- 适用于大文件高并发的环境



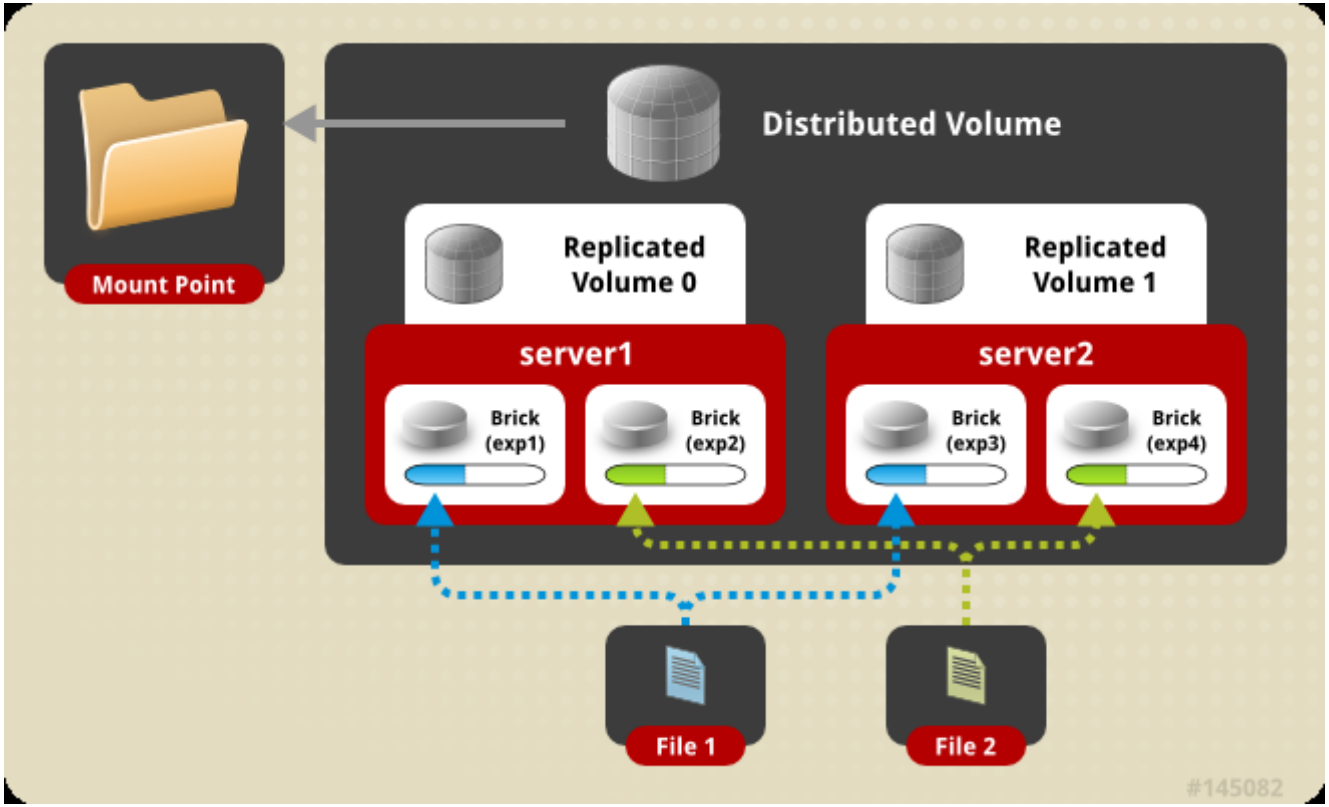
volume 类型 - 分布条带式

- 分布式和条带式的组合
- 适用于大文件高并发同时又能扩展容量的环境



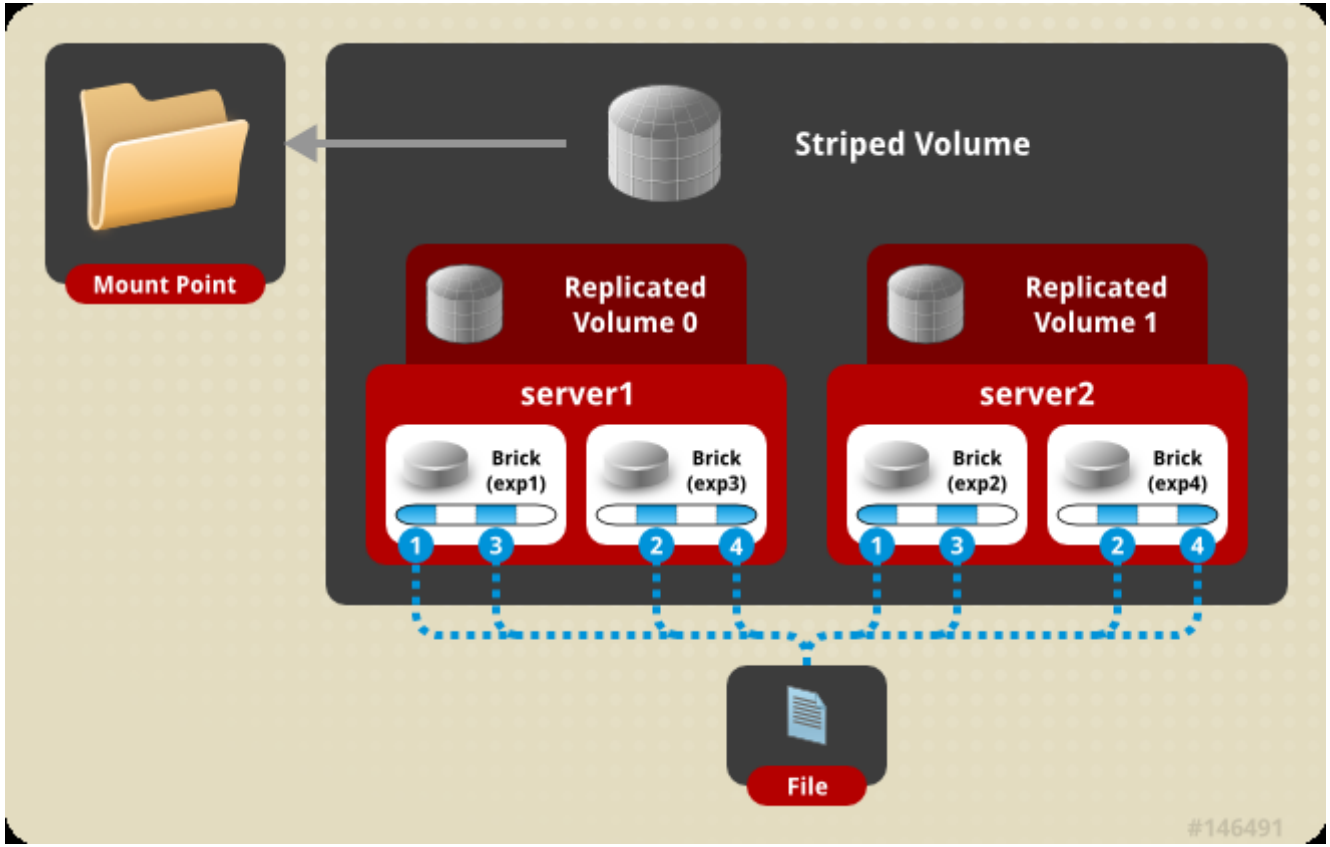
volume 类型 - 分布复制式

- 分布式和复制式的组合
- 适用于高可靠又能扩展容量的环境



volume 类型 - 条带复制式

- 分布式，分布式和复制式的组合
- 适用于高可靠，高并发的环境



volume挂载类型

- gluster native client

  glusterfs提供的linux fuse客户端，安装完成后适用mount命令挂载

  mount -t glusterfs HOSTNAME-OR-IPADDRESS:/VOLNAME MOUNTDIR

- nfs挂载

mount -t nfs -o vers=3 HOSTNAME-OR-IPADDRESS:/VOLNAME MOUNTDIR

- CIFS

  windows上的挂载方式

使用步骤

- 安装glusterfs

  centos上使用yum或者从源码安装

- 启动glusterd服务

  /etc/init.d/glusterd start

- 选择一种类型创建glusterfs volume(glusterfs 逻辑卷)

  使用glusterfs 控制台命令

- 启动逻辑卷

  使用glusterfs 控制台命令gluster volume start test-volume

- 挂载逻辑卷

  选择某种方式挂载后即可像本地磁盘一样访问

how it works　各个程序作用

- gluster

  glusterfs 命令行工具

- glusterd, glusterfs, glusterfsd

  同一个程序，根据配置文件的不同执行不同的功能

  glusterd： glusterfs的管理程序，主要执行来自gluster的命令

  glusterfsd: glusterfs各个存储节点的daemon程序，用于管理某个存储节点， 每个节点会运行一个该daemon程序

  glusterfs： Glusterfs 的fuse客户端程序，该程序与每个glusterfsd都会建立一个链接

how it works　xlator

- glusterfs软件采用模块化的结构，每一个模块就是一个xlator
- 每一个xlator可以单独进行配置
- glusterfs，glusterd，glusterfsd启动时，根据配置文件的不同生成不同的xlator的树形结构，实现不同的功能，程序自上而下执行各个xlator
- 每个xlator就是一个so文件

/usr/lib64/glusterfs/3.3.1/xlator/cluster/distribute.so -> dht.so

/usr/lib64/glusterfs/3.3.1/xlator/cluster/replicate.so -> afr.so

/usr/lib64/glusterfs/3.3.1/xlator/debug/io-stats.so

how it works　vol file

- vol file是glusterfs程序的配置文件，跟xlator对应

glusterd　/etc/glusterfs/glusterd.vol

glusterfsd　/var/lib/glusterd/vols/pubbak/, 每个brick对应一个vol file

glusterfs　pubbak-fuse.vol

运维经验

- glusterfs 并不完善，当出现问题是需要查看glusterfs的日志，/var/log/glusterfs/*, 每个brick有单独的日志文件
- 根据日志文件的提示，大部分问题可以从网上搜到解决方法
- 部分问题需要查看源代码，找到work around的方法
- 有一部分可能需要修改源代码，要和社区紧密合作

常见问题分析　脑裂(brain split)

- 现象

  复制的两份文件数据或者扩展属性不一致

- 原因

  一般是由于某台服务器意外重启造成的

- 解决方法

  1. 确定数据或者扩展属性是坏的那份copy, 通过 /export 挂载点将其从磁盘上删掉

  2. 通过glusterfs挂载点ls该目录或者文件，glusterfs自身的修复机制会重新建一份新的copy

常见问题分析 ls慢

- 现象

  当目录下有大量文件或者目录时，ls时间很慢，90000多的目录需要长达10分钟左右的时间才返回

- 原因

  慢的原因主要是glusterfs没有统一的元素据管理，使得ls执行时两个系统调用getdents 和 lstate操作都比较慢，glusterfs 3.5做了一些优化，提供了performance.readdir-ahead和performance.force-readdirp两个优化选项， 能提高这两个系统调用的执行时间，在测试系统ls同一个目录能有一倍左右的提高

- 解决方法

  ls -1 -U --color=never

  这个ls命令不会调用lstate，但无法看到size, type之类的属性

常见问题分析　扩容后rebalance慢

- 现象

  当把一个新的brick加入集群时需要手动执行rebalance命令，glusterfs会移动数据到新的brick上，我们的一次rebanlance的过程将近一个月才完成(不影响使用)

- 原因

  原因与我们的应用有关系，我们的存储中有海量的小文件，而且每个文件在一个单独的目录下，这种特殊的结构与glusterfs的设计不符。每一个目录，glusterfs会在所有的节点上都创建。这个过程的耗时远大于真正的数据传输。

常见问题分析　brick再加入问题

- 现象

  gluster volume delete pubbak删除一个逻辑卷，再将该逻辑卷的brick重新加到一个集群时，日志出现如下错误：

/export/sdb1 or a prefix of it is already part of a volume

- 原因

  这是因为gluster volume delete pubbak病没有清除brick挂载点的扩展属性，导致不能再加入

- 解决方法

  手动清除扩展属性:

  setfattr -x trusted.gfid /export/sdb1

  setfattr -x trusted.glusterfs.volume-id /export/sdb1

常见问题分析　跨机房复制(geo replication)

- 什么时候需要跨机房复制

一个glusterfs集群通常部署在一个机房里，当机房断电时导致服务不可用，需要在另一个机房有个备份的集群，跨机房复制在配置好后能自动将主集群上的数据同步到从集群，对应用层透明

- 跨机房复制原理

跨机房复制是有一个单独的python层序完成，叫gsync, 最底层使用rsync进行数据传输。

- gsync与rsync同步比较

rsync默认会检查每一个文件的大小和修改时间，当这两个属性有一个不一样时， rsync会将其列入到传输的文件列表中进行传输，对于拥有海量小文件的存储来说，在两边一样的情况下跑一次rsync需要几天的时间

gsync的检查机制完全不一样，利用了glusterfs的xtime属性和目录结构，如果目录的xtime属性一样，gsync会直接忽略掉该目录下的所有子目录和文件。我们的目录结构是以日期为单位，以往目录的xtime属性都一样，直接被忽略掉，这样很快就能检查到增量的部分。在两边一样的情况下瞬间就能完成一次同步

常见问题分析　跨机房复制(geo replication)

- 跨机房复制还不是很完善，使用过程中碰到了很多，极大增加了运维的工作
- 常见问题

1. 当主集群已经有海量数据，在新增加从集群时，geo-replication误报大量的错误日志，需要脚本及时清理

2.从集群上误删文件后，不能被发现重新同步(父目录的xtime属性没变)

3. 从glusterfs卷停止后重启不了：

volume start: pubbak: failed: Volume id mismatch for brick

需要复杂的work aroud方法

4. 各种异常情况层出不穷

- 解决办法

在使用了几个月后，还有各种新的问题出现，于是停止使用这个功能，应用层自己负责写两个集群，运维工作剧减

常见问题分析　nfs内存泄露问题

- 现象

使用glusterfs一段时间后出现服务器内存报警(3.3版本)，检查发现glusterfs nfs deamon进层内存使用极高

- 原因

没有进一步检查

- 解决方法

由于我们并没有使用nfs挂载的方式，直接用nfs.disable将nfs deamon禁掉

常见问题分析　rpc调用内存泄露问题

- 现象

360作为安全公司会经常对自己的服务器进行安全攻击，在一次伪造rpc攻击中glusterfs集群中的服务器内存急剧升高

- 原因

glusterfs对某些类型的rpc支持不好，接收到该类型的rpc调用时会进入分配内存的死循环中，很快耗光内存

- 原因

提交bug和解决方法给社区，新版已采纳并修改

总结

- 虽然状况不断，但一个个解决后，现在线上集群跑得很好
- 比较健壮，在各种问题中还能保持正常服务
- 社区活跃，在持续不断完善中