分层存储集成到OSS设计

Created and last modified by 肖飞 on Nov 25, 2015

功能模块划分

- HUB
  维护OSS上关于SubPool、Disk、OBJ配置信息以及OBJ IO信息，并负责和AST（分层存储）交互。
- ASTIOModule
  截获关于OBJ的创建/删除事件，并汇报给HUB；
  截获关于Disk的添加/删除事件，并汇报给HUB；
  截获关于SubPool的添加/删除事件，并汇报给HUB；
  截获关于OBJ的访问事件，更新OBJ IO信息；
- MIG
  处理迁移请求，在迁移完成时回复AST迁移完成消息并通知MDS更新元数据。

HUB

- HUB启动时机
  HUB启动时机有两种选择：
  - 在Enable AST的时候启动
    - HUB在Enable AST的时刻需要扫描OSS上所有的Disk，以便建立逻辑配置视图，扫描过程会比较费时，而且如果用户频繁的Enable AST然后Disable AST，则会触发频繁的扫描。
    - HUB在Disable AST的时刻销毁逻辑配置视图。
  - 随OSS启动而启动
    - HUB在OSS启动时刻开始建立逻辑逻辑配置视图，并在OSS运行过程中更新逻辑配置视图，但是只在Enable AST后才开始进行IO统计，Disable AST后停止进行IO统计（Disable AST是高优先级事件，优先处理）。
    - HUB在OSS停止时候销毁逻辑配置视图。
  - 目前首选方案2.
- 配置信息和IO信息维护
  - 解析ASTIOModule关于Disk添加事件，建立Disk添加时刻的逻辑配置视图；
  - 解析ASTIOModue关于Disk删除事件，删除该Disk的逻辑配置视图；
  - 解析ASTIOModule模块关于OBJ/Disk/SubPool的添加/删除事件，更新OSS运行过程中逻辑视图；
  - 对于Disk添加事件的处理，不仅仅要添加Disk到逻辑配置视图中，还要将Disk添加时刻扫描到的初始OBJ集合添加到逻辑配置视图中（此时要注意加锁保护，否则可能存在多线程访问问题）；
  - 对于Disk添加事件的处理，还要判断该Disk相关的SubPool是否存在，如果不存在，则添加该SubPool到逻辑配置视图中；
  - 对于Disk删除事件的处理，需要删除该Disk中所有OBJ的配置；
  - 对于Disk删除事件的处理，还需要判断该Disk是否是它所对应的SubPool中的最后一个Disk，是则删除对应的SubPool；
  - SubPool添加事件派生自Disk添加事件；
  - SubPool删除事件派生自Disk删除事件；
- 和AST交互
  - 专门开启一个线程负责和AST交互；
  - 接收AST发送过来的关于OSS逻辑配置请求，响应为OSS当前逻辑配置视图；
  - 接收AST发送过来的关于OBJ IO信息请求，响应为OSS当前逻辑配置视图中所有OBJ的IO信息；
  - 接收AST发送过来的迁移请求，并进一步传递给MIG处理；
  - 主动push OSS中关于SubPool/Disk/OBJ的删除事件给AST；

ASTIOModule

- 添加ASTIOModule类

  在StorageManager::Start中添加StartASTIOModule；

  在ASTIOModule中实现以下函数：

  Initialize --- 其中需要维护IOModules链表；

  Finalize --- 需要链式删除后续IOModule；

  Create --- 创建对象

  Replicate --- 到新的节点或者硬盘上复制对象

Delete --- 删除对象

Write --- 写操作统计

Read --- 读操作统计

其它函数实现都是直接调用GetNextIOModule()->***(cfh, param);

- 关于OBJ添加/删除事件

  OBJ添加/删除事件的监控需要在AST IO Module中截获create、delete、replicate事件，其中create和replicate事件都当做添加事件来处理，但是replicate的时候需要区分是因为disk删除导致的还是snapshot导致的，对于snapshot导致的，暂时不管，这种类型的snapid不为0。
- 关于Disk/SubPool的添加/删除事件
  - 关于Disk添加事件，可以在ASTIOModule的Initialize中进行，因为在StorageManager启动各IOModule之前都已经确保scan结束了，直接发送Disk添加事件给OSS即可，事件中要包含DiskKey和DiskBaseProp信息。DiskKey可以通过StorageManager获取，DiskBaseProp则需要想办法。（可能需要小明在资源管理中添加DiskBaseProp相关的信息的获取，并在Monitor传递Disk信息给OSS时候附带上这些信息）。
  - 关于Disk删除事件，可以在ASTIOModule的Finalize中进行。
  - 关于SubPool添加事件，不用单独为SubPool产生添加事件，而是作为Disk添加事件的派生事件，直接在HUB中处理。SubPoolKey所需要的subpoolid可以直接从StorageManager：：GetPoolId获取，所需要的ossid可以通过ControlCenter -> StorageServiceMonClient::GetSid获取；
  - 关于SubPool删除事件，不用单独为SubPool产生删除事件，而是作为Disk删除事件的派生事件，直接在HUB中处理。SubPoolKey的获取同上。
- 关于OBJ的访问统计

对于Read、Write涉及到的OBJ直接发送OBJ访问事件给HUB，在HUB中进行IO模式等的计算和访问次数等的计算，尽可能减少在IOModule中的时间消耗。其中OBJ访问事件中会包括read/write, offset, len等信息。

MIG

- 迁移模块的实现依赖于OSS中的Reselect Path逻辑
  - 直接构造消息，然后调用ControlCenter：：ForwardToObjectLayout即可。构造消息参考MetaProtocolParser::ReselectPathMessageToContext，该函数中需要从NMsgBlock中获取信息填充FS::Object::Migrate结构，但是我们构造消息的时候没有NMsgBlock，所以需要自己调用PB相关的接口去填充FS::Object::Migrate。
  - 对于AST发送过来的迁移请求需要区别于正常的reselect path消息，因为在迁移完成之后需要发送完成事件给HUB，进而转发给AST。可以在OpCtx中指定自己实现的继承自CommDispatcher的类负责分发，如ASTCommDispatcher。
  - 但是由于当前OSS设计中要求CommDispatcher有自己的线程，如果不希望开启新的线程，可以直接将HUB继承自CommDispatcher，然后将HUB和ASTCommDispatcher 合并到同一个类中，但是目前首选为ASTCommDispatcher开启自己的线程。
- 迁移模块要能够感知系统中OBJ是否还存在，目前OSS 的reselect的实现已经保证
- 迁移完成之后需要汇报MDS关于对象位置更新信息
  - 直接复用OpCtx，并设定其中mDstSID和mDispatcher，其中mDstSID设定为迁移后Chunk所在的MasterMDs，mDispatcher设定为CommunicationLayout::mMetaDispatcher。
- 迁移完成后需要更新HUB中维护的关于OBJ的配置和IO信息
  - 直接向HUB发送事件即可