SoftSAN系统设计

注：本文内容均为主观臆断的想法，请在实践中修订完善

**一、系统架构**

如图1所示，SoftSAN系统组成包括集中式Meta-Data Server（MDS，元数据服务器）、ChunkServer（数据服务器）和Client（客户端），其中ChunkServer和Client角色可以由同一节点担当。系统中各个节点之间的通信通道采用消息队列服务，通信协议采用Google的Protocol Buffer（PB）技术制定。SoftSAN系统中的3大组成部分与GFS中的对应部分具有十分类似的功能，具体细节可查阅资料并进行对比。



图1 SoftSAN系统架构

MDS的功能包括：1.记录系统中包含的卷列表（即卷目录），卷目录组成树形层次结构，与常规的文件系统结构类似；2.对于每个卷保存其包含的数据块（Chunk）列表；3. 对于每个卷保存数据块的组装结构、组装方法和组装参数；4.记录系统中每个节点的状态，包括存活情况、性能容量状态、健康状态等；5.在出现故障时协调各个其他节点完成故障恢复任务。6.MDS必须支持高可用（HA）技术，如双机热备。

ChunkServer功能包括：1.创建并导出数据块，数据块可存在本地文件系统上，也可以存在LVM之上；2.监测服务节点健康状态；3.在MDS的指导下做故障恢复。

Client功能包括：1.创建、修改、维护、装配、删除卷；2.分为命令行工具与运行时库两部分，分别供管理员和管理程序或应用程序使用。

**二、概要设计**

**1. Meta-Data Server**

如图2所示，MDS的主要组成模块包括Main、MetaDataServer、MessageQueue、TransientDB、PersistentDB、PyRedis和FileCDP，其中PyRedis是现有模块，由Redis提供，负责与Redis服务器通信。

FileSystemCDP：提供文件级的CDP（Continuous File Protection，持续数据保护）功能，其总体目标是允许管理员在一定范围内撤销对卷的任何正常操作。由于每个卷的信息存储为一个文件，卷目录存储为一个文件目录，因而对卷的CDP即对文件的CDP。对于一个文件”asdf”，创建一个对应的zip文件”.asdf”，以存放该文件的历史内容；每当要对文件实施变更操作前，先将该文件存入对应的zip文件中，并将zip包中的新增文件名设为”asdf.当前时间” 。每目录内设置特殊文件“.\_ \_dirinfo\_ \_”（“\_ \_”为连续双下划线），记录目录内的变更，包括且仅包括文件和子目录的创建、删除、更名。

PersistentDB：负责记录系统中各种持久信息，如卷包含哪些数据块，数据块如何装配等信息。该模块在实现层次面上对*FileSystem*进行封装，并从卷管理的角度提供更高的抽象层次，对外暴露各种卷管理层面的方法、函数。

HistoryView：是FileSystemCDP的从属模块，用于表示受护文件系统在某个历史时刻的状态，时间点可以动态设置。HistoryView与FileSystemCDP具有共同的父类*FileSystem*，因而可向PersistentDB提供历史状态的访问能力。此外HistoryView还将在*FileSystem*的基础上拓展出针对历史数据的查找、恢复的功能，因而PersistentDB必须能够直接访问本模块，不能仅仅通过*FileSystem*基类。



图2 MDS主要组成模块与静态结构

TransientDB：负责记录各种短时状态，如各个节点的在线情况、健康状态等。该模块通过PyRedis使用Redis记录、查询数据。

MessageQueue：是SoftSAN系统中各个节点间的通信通道，通过封装PyRedis中的pub/sub消息队列功能实现，主要是提高抽象层次。

MetaDataServer：从整体上负责MDS的全部功能性事务，其工作流程大致为从MQ接收请求🡪在TransientDB和PersistentDB的支持下执行请求🡪返回结果。MDS不设置退出方法，其终止途径为kill或^C。

Main：可理解为MDS的外皮，负责解析命令行参数，创建MDS执行所需的模块对象，并配置初始参数。

**2. ChunkServer**

如图3所示，ChunkServer的主要组成模块包括：Main、ChunkServer、*Frontend*和*Backend*。其中*Frontend*和*Backend*是抽象基类，其功能分别为Chunk服务前端和Chunk存储后端。前端的具体类包括Frontend\_tgt和Frontend\_LIO，分别控制tgt和LIO这两种iSCSI服务器。目前ChunkServer只支持iSCSI（SCSI over IP）服务协议，以后可能会扩充FCoE（Fiber Channel over Ethernet）、AoE（ATA Channel over Ethernet）等协议。ChunkServer的功能与OpenStack项目中的nova-volume服务具有一定的相似性，可参考nova-volume的实现，甚至借鉴一部分代码。

*Frontend*：目前只需要实现tgt的控制器，以及nova-volume支持的其他控制器，等RHEL/CentOS完全支持LIO后再实现LIO控制器。

*Backend*：本版本应同时支持FS后端和LVM后端，并且支持Thin-Provisioning（以下简称thin）。Thin的意义在于分配Chunk存储对象时，并不立刻分配存储空间，而是推迟到真正写入数据时才分配。对于FS后端可采用稀疏文件技术实现，LVM的该功能还处于beta测试阶段，可暂不实现。Backend的Chunk创建函数应留出一个参数指明是否创建thin Chunk，该参数只作为建议值，不强制。



图2 ChunkServer主要组成模块与静态结构

PyLVM：是已有模块，负责与LVM系统交互。

ChunkServer：从整体上负责Chunk服务器的全部功能性事务，其工作流程大致为从MQ接收请求🡪在*Frontend*和*Backend*的支持下执行请求🡪返回结果。ChunkServer不设置退出方法，其终止途径为kill或^C。

Main：可理解为Chunk服务器的外皮，负责解析命令行参数，创建Chunk服务器执行所需的模块对象，并配置初始参数。

**3. Client**

SoftSAN用户使用Client对卷实施各种操作，当前版本不区分用户和权限。Client分为两部分Client-API和Client-CLI，其中前者在后台运行，完成各种实质性工作，并持续监控已挂载卷的状态，响应内核模块的各种协助需求；后者是供用户使用的命令行工具，通过调用前者完成用户命令。Client-API和Client-CLI通过MQ和GPB通信，与其构件间通信方式类似。

如图3所示，Client-API的主要组成模块包括：Main、Client、SoftSAN、Volume、PyRedis和PyDM，其中PyRedis和PyDM分别用于封装Redis和Device Mapper的API。

SoftSAN：是Client端对SoftSAN实施操作的核心模块，它对外提供了各种功能的API，装化为对系统内部实施的各种微操作。

Volume：表示SoftSAN系统中的一个卷。卷聚合结构为又向无环图，通常（但不一定）只有一个0出度节点（终结点）。若系统只有1个终结点且其他所有节点出度均为1，则系统拓扑退化为树（多数情况应该如此）。关于卷结构的详细论述见下一小节。

Client：负责接收用户输入，调用SoftSAN的相应功能，返回执行结果。

Main：负责解析命令行参数，创建执行所需的模块对象，并配置初始参数。可根据实际情况与Client合并。



图3 Client-API主要组成模块与静态结构

**4. 卷的抽象表示结构**

先看看常见应用中的卷结构：



(a) 类似GFS的卷结构



(b) 支持虚拟机运行的卷结构



(c) 具有代理节点的卷结构



(d) 带有重排序机制的卷结构

图4 多种常见卷结构

如图4所示，在上述4种常见存储结构中，前3种都可以用树形描述，而第4种则不可以，因而我们采用更具有普适性的有向无环图结构来描述，其中方向定义为服务提供方向，即从服务器指向客户端。

卷的数据结构（有向无环图）定为每个节点保存其前驱节点列表以及可选的边参数，对于图4里的前3种结构，可退化为树形结构表示；对于第4种结构，其关键点在于将一个存储块节点切分为多个，此时每个子节点均保存共同的前驱节点，而边参数保存其在前驱中的切分位置，是互不相同的。

系统给每个卷都分配一个全局唯一的128位ID（Guid），包括卷的子节点以及Chunk都视为特殊类型的卷。