1. Blobstore
   1. blobstore介绍

Blobstore被定义为一种块分配的存储池，从层次上讲，blobstore是构建在底层块设备上的一种逻辑块分配的存储系统，类似于传统意义上的文件系统，但并非文件系统。Blobstore也不支持posix语义，同时为了和传统posix语义上的文件系统区分开，blobstore使用了blob对象来表示传统文件系统中的files和objects对象。

从应用上，blobstore为上层的应用提供了基础的块存储切片服务，从而可以基于blobstore构建很多现有的上层存储服务。例如：逻辑卷、数据库、文件系统（BlobFS和RocksDB等）、key/value存储等等，甚至包括SAN、NAS和各类分布式存储系统。

* 1. blobstore概念

Blobstore建立了抽象的层次式存储结构，其中包含的对象有：blobstore、blob、cluster、page以及logical block（逻辑块）。

逻辑块：底层块设备的最小访问单元，一般512B或4KB

Page页：包含了固定的逻辑块数目，page中的逻辑块一定是物理连续的。Page大小一般4KB，SSD设备必须能够原子的读写一个Page。

Cluster簇：包含了固定数目的Page，这个数目是创建blobstore时，指定的。一般一个cluster大小1MB或者4MB。Cluster内的Page也是物理上里西安许的。

Blob：blob是一个有序簇组成的簇链表，应用程序会直接操作blob。Blob是物化到块设备的，因此断电和重启数据不丢失。应用程序通过blobstore中保存的blob标识符来访问特定的blob，blob的读写访问以page为单位，通过指定page在blob中的起始偏移即可。应用程序可以往blob中存储key/value键值对，这些键值对实际形成了blob的属性（xattrs）。

Blobstore: 一个blobstore管理了整个底层块设备，基于blobstore的应用程序，被称作一个blobstore。Blobstore由元数据区域和多个blobs集合组成。

下图描述了blobstore中的逻辑层次结构：



1. blobstore的关键数据结构
   1. spdk\_blob\_store

struct spdk\_blob\_store {

uint64\_t md\_start; /\* Offset from beginning of disk, in pages \*/

uint32\_t md\_len; /\* Count, in pages \*/

struct spdk\_io\_channel \*md\_channel;

uint32\_t max\_channel\_ops;

struct spdk\_thread \*md\_thread;

struct spdk\_bs\_dev \*dev;

struct spdk\_bit\_array \*used\_md\_pages;

struct spdk\_bit\_array \*used\_clusters;

struct spdk\_bit\_array \*used\_blobids;

pthread\_mutex\_t used\_clusters\_mutex;

uint32\_t cluster\_sz;

uint64\_t total\_clusters;

uint64\_t total\_data\_clusters;

uint64\_t num\_free\_clusters;

uint32\_t pages\_per\_cluster;

spdk\_blob\_id super\_blob;

struct spdk\_bs\_type bstype;

struct spdk\_bs\_cpl unload\_cpl;

int unload\_err;

TAILQ\_HEAD(, spdk\_blob) blobs;

TAILQ\_HEAD(, spdk\_blob\_list) snapshots;

};

* 1. spdk\_bs\_super\_block

struct spdk\_bs\_super\_block {

uint8\_t signature[8];

uint32\_t version;

uint32\_t length;

uint32\_t clean; /\* If there was a clean shutdown, this is 1. \*/

spdk\_blob\_id super\_blob;

uint32\_t cluster\_size; /\* In bytes \*/

uint32\_t used\_page\_mask\_start; /\* Offset from beginning of disk, in pages \*/

uint32\_t used\_page\_mask\_len; /\* Count, in pages \*/

uint32\_t used\_cluster\_mask\_start; /\* Offset from beginning of disk, in pages \*/

uint32\_t used\_cluster\_mask\_len; /\* Count, in pages \*/

uint32\_t md\_start; /\* Offset from beginning of disk, in pages \*/

uint32\_t md\_len; /\* Count, in pages \*/

struct spdk\_bs\_type bstype; /\* blobstore type \*/

uint32\_t used\_blobid\_mask\_start; /\* Offset from beginning of disk, in pages \*/

uint32\_t used\_blobid\_mask\_len; /\* Count, in pages \*/

uint8\_t reserved[4012];

uint32\_t crc;

};

* 1. Spdk\_blob\_mut\_data

struct spdk\_blob\_mut\_data {

/\* Number of data clusters in the blob \*/

uint64\_t num\_clusters;

/\* Array LBAs that are the beginning of a cluster, in

\* the order they appear in the blob.

\*

\* blob中每一个簇的lba起始位置

\*

\*/

uint64\_t \*clusters;

/\* The size of the clusters array. This is greater than or

\* equal to 'num\_clusters'.

\* 上述数组大小

\*/

size\_t cluster\_array\_size;

/\* Number of metadata pages

\*

\* blob的元数据pages数目

\*

\* \*/

uint32\_t num\_pages;

/\* Array of page offsets into the metadata region, in

\* the order of the metadata page sequence.

\*

\* blob元数据占用的page数组，lba表示

\*

\*/

uint32\_t \*pages;

};

* 1. spdk\_blob

struct spdk\_blob {

struct spdk\_blob\_store \*bs;

uint32\_t open\_ref;

spdk\_blob\_id id;

/\*

\*

\* 父blob

\*/

spdk\_blob\_id parent\_id;

enum spdk\_blob\_state state;

/\* Two copies of the mutable data. One is a version

\* that matches the last known data on disk (clean).

\* The other (active) is the current data. Syncing

\* a blob makes the clean match the active.

\*/

struct spdk\_blob\_mut\_data clean;

struct spdk\_blob\_mut\_data active;

bool invalid;

bool data\_ro;

bool md\_ro;

/\*

\*

\* blob读写相关的标志位

\*/

uint64\_t invalid\_flags;

uint64\_t data\_ro\_flags;

uint64\_t md\_ro\_flags;

struct spdk\_bs\_dev \*back\_bs\_dev;

/\* TODO: The xattrs are mutable, but we don't want to be

\* copying them unecessarily. Figure this out.

\*/

struct spdk\_xattr\_tailq xattrs;

struct spdk\_xattr\_tailq xattrs\_internal;

TAILQ\_ENTRY(spdk\_blob) link;

};

1. Blobstore的物理存储
2. Blobstore中的元数据
   1. 全局blobstore的元数据
   2. 每一个blob的元数据