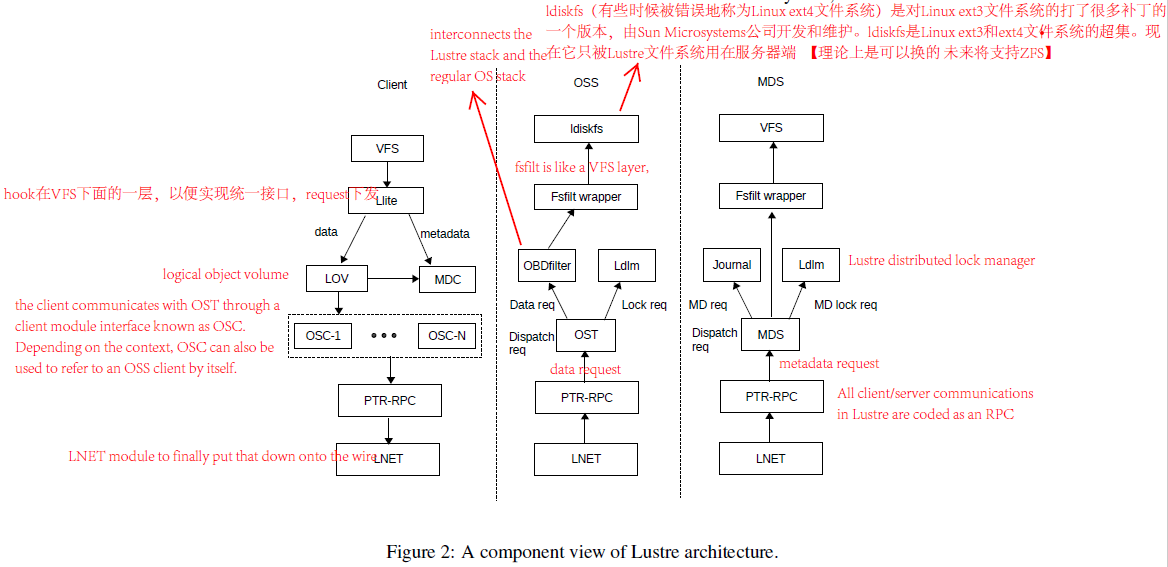
**Understanding The Lustre Filesystem Internal阅读笔记**

**1 Component View on Architecture**



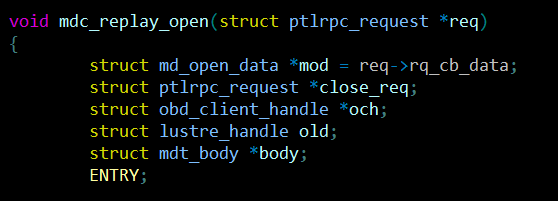
Llite是负责衔接VFS和Lustre，提供统一访问接口的中间层。

|  |  |
| --- | --- |
|  | 图示代码就是这个衔接的过程，具体体现在  lustre/llite文件夹下的代码中  尤其是file.c 和dir.c |

|  |  |
| --- | --- |
| **dir.c** | **衔接过程** |
|  |  |

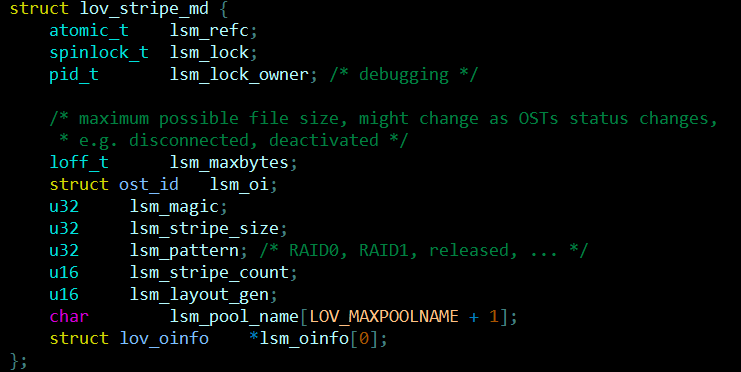
|  |  |
| --- | --- |
| **file.c** | **衔接过程** |
|  |  |

元数据存储在MDS，与元数据相关的操作例如create，open，read等操作，都与MDS相关，这是通过MDC模块实现的。具体的是ptlrpc\_request，因此ptlrpc\_request是核心，例如下面的操作（lustre/mdc/mdc\_request.c）：



一个对象文件的布局信息存在inode的extended attribute（EA）中，所以如果分条大小是1MB，那么这意味着[0, 1M), [3M, 4M) …作为对象x存储在OST p；[1M, 2M), [4M, 5M) …作为对象y存储在OST q; [2M,3M), [5M, 6M) …作为对象z存储在OST r。

在读取文件之前，客户端将首先通过MDC询问MDS，从而得知它应当就这个操作，和<ost p, ost q, ostr>进行对话。这些信息组织在所谓的LSM中，而客户端的LOV（logical object volume，逻辑对象卷）则是用来解释这个信息的，这样就使得客户端能够向OST发送请求。需要重申的是，客户端通过一个称为OSC的客户端模块接口，和OST进行通信。根据上下文的不同，OSC也可以用来指称OSS客户端。



|  |
| --- |
| /\*\*  \* File IDentifier.  \*  \* FID is a cluster-wide unique identifier of a file or an object (stripe).  \* FIDs are never reused.  \*\*/  struct lu\_fid {  /\*\*  \* FID sequence. Sequence is a unit of migration: all files (objects)  \* with FIDs from a given sequence are stored on the same server.  \* Lustre should support 2^64 objects, so even if each sequence  \* has only a single object we can still enumerate 2^64 objects.  \*\*/  \_\_u64 f\_seq;  /\* FID number within sequence. \*/  \_\_u32 f\_oid;  /\*\*  \* FID version, used to distinguish different versions (in the sense  \* of snapshots, etc.) of the same file system object. Not currently  \* used.  \*\*/  \_\_u32 f\_ver;  }; |