4、Malloc Library

Librte\_malloc库提供API可以实现任意大小的内存分配。

该库的目的是提供类似malloc的内存分配函数用于从hugepage内存中分配内存，从而提高程序的可移植性。API详见 The DPDK API Reference.

 注意，该库提供的分配任意大小内存的函数，分配的过程中会使用到lock，并且分配的效率比pool-based的分配方式低，因此不应该在数据包处理过程中调用；当然，可以在配置平面中使用。

可以参考 DPDK API Reference手册中对rte\_malloc函数的详细描述。

4.1 Cookies

当CONFIG\_RTE\_MALLOC\_DEBUG宏开启时，分配函数开启了内存的写保护机制以帮助排查缓冲区溢出的问题。

4.2 对齐与NUMA限制

Rte\_malloc() 中有一个参数指定了内存对齐最小单元（该最小单元必须是2的整数次幂）

Rte\_malloc()函数分配的内存是从分配核亲和内存中获得。当然，DPDK也提供了从其他核的亲和内存中分配内存的API。

4.3 使用场景

librte\_malloc库适用于在初始化或配置平面中需要做类似malloc的内存分配时使用，通过该库分配的内存返回地址只返回逻辑地址，不返回物理地址。

在数据处理平面中（运行时，快路径），分配和释放内存应该使用memory pool library。

4.4内部实现

4.4.1 数据结构

Malloc library 在内部使用了两个重要的数据结构

Struct malloc\_heap 用于跟踪每个核上的空闲内存

Struct malloc\_elem 已分配的和空闲元素总表

4.4.1.1 malloc\_heap

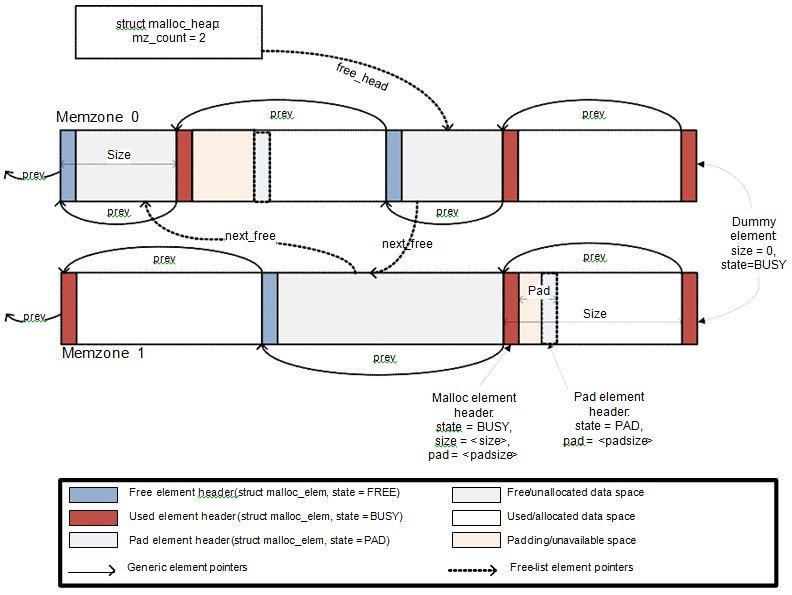
malloc\_heap 用于跟踪每个核上的空闲内存。在库的内部，每个NUMA节点上都有自己的heap结构体，这就允许在线程分配其运行核上的内存。当然，分配的内存其他的核也可以使用，这种分配方式不会比总是在一个固定核上分配内存使用效率更差。  
malloc\_heap主要的成员如下：

mz\_count：用于计算这个核上有多少memory zones分配给了heap memory，这个值主要的使用场景是可以和numa\_socket 值一起为每个memory zone命名时生成唯一的名字

lock：用于同步heap的操作。Heap中的空闲空间有一个链表串连着着，需要防止两个线程同时对该链表进行操作。

Free\_head：该指针指向空闲结点的链表头部。

**Figure 3. Example of a malloc heap and malloc elements within the malloc library**



4.4.1.2 structure: malloc\_elem

Malloc\_elem是适用于各种内存块的通用的头结构体，有三种使用方式：

1. 作为一块已分配或者未分配内存的头部（一般方式）
2. 作为一块内存的填充头部
3. 作为一个memzone的结尾标识