**内核学习笔记-文件系统篇**

如果操作系统的职责“管理和抽象硬件资源并为上层应用提供”来解释文件系统的话，文件系统就是管理并抽象存储资源（磁盘、Flash等），完成从应用层到物理存储介质的IO（申请、读写、擦除、释放等），并针对不同存储介质特点对IO管理起到性能、损耗、安全等优化效果。

# 基础知识

## 文件系统分层

Linux设计哲学是将将一切都看作文件，普通文件（包括文本和二进制）、目录文件、字符设备文件、块设备文件等。Linux文件系统的层次如下所示：



各层的主要功能职责为：

|  |  |  |
| --- | --- | --- |
|  | 层名 | 功能职责 |
| 用户态(User Space) | 应用层[Application] | 基于标准库，构建应用程序 |
| C标准库[Standard Lib C] | 基于系统调用接口，构建Linux平台的C标准库，提供上层应用开发的统一编程接口 |
| 内核(Kernel) | 虚拟文件系统层[VFS] | 基于不同文件系统的调用接口，抽象不同文件系统的资源管理方式，以inode构建通用文件模型，管理文件数据和超级块的映射方式，提供对用户态提供统一的系统调用接口 |
| 文件系统层 | 基于通用的块层资源访问接口，组织文件和逻辑块层的映射存储关系，将文件读写、新建、删除操作转换成对应的逻辑块读写、申请、释放操作，提供对应文件系统的读写、新建、删除操作接口 |
| 块层[General Block Device Layer] | 基于不同磁盘的驱动接口，将磁盘物理块抽象为逻辑块，并负责从逻辑块到物理块的映射。管理逻辑块资源，为上层文件提供统一的块操作接口 |
| 驱动层[Device Driver] | 基于实际物理资源，不同厂家提供了不同的磁盘操作接口 |
| 物理(Hardware) | 磁盘层[Physical Disk] | 物理磁盘，提供真实的存储资源 |

以Ext2文件系统写入数据到文件为例讲述从上到下的整个流程：

## 区分块、卷、分区、扇区

逻辑分区 VS 物理磁盘：



|  |  |
| --- | --- |
| 名称 | 含义 |
| 物理分区[PP (Physical Part)] | 代表一块物理磁盘 |
| 扇区[Sector] | 物理磁盘写入、读取的最小单位，由若干地址连续的存储单元组成，一般为512B |
| 物理块[PE (Physical Extends)] | 由若干连续的扇区组成，一定是块大小的2^N倍 |
| 逻辑块[LE (Logical Extends)] | 和物理块大小相同，可以通过映射得到对应的物理块。操作系统读写磁盘的基本单元是块 |
| 逻辑分区[LP (Logical Part)] | 由若干逻辑块组成（不要求逻辑或物理地址连续），每个分区可拥有各自的文件系统，对整个磁盘进行划分，更合理地使用硬件资源 |

相比于物理磁盘，逻辑分区允许一块磁盘上运行不同的文件系统，使得硬件资源利用率更高。可以发现，逻辑块和物理块大小相同，逻辑块资源总和应该和物理磁盘大小相同，那么为什么要进行一个从逻辑块到物理块的映射？

如果没有逻辑抽象那么所有的写操作则只支持顺序，逻辑块主要为了支持物理磁盘的随机读写，上层应用完全可能修改某个已经落盘的文件，使得对应的物理块大小发生变化（可能需要申请新物理块或释放旧物理块），这类似于物理内存随机读写的技术实现。

逻辑卷 VS 逻辑分区：



|  |  |
| --- | --- |
| 名称 | 含义 |
| 物理卷[PV (Physical Volume)] | 代表一块物理磁盘 |
| 扇区[Sector] | 物理磁盘写入、读取的最小单位，由若干地址连续的存储单元组成，一般为512B |
| 物理块[PE (Physical Extends)] | 由若干连续的扇区组成，一定是块大小的2^N倍 |
| 容器池[Volume Group] | 将多块磁盘的物理块组合，并抽象为一个容器池，里面提供很多逻辑块 |
| 逻辑块[LE (Logical Extends)] | 和物理块大小相同，可以通过映射得到对应的物理块。操作系统读写磁盘的基本单元是块 |
| 逻辑卷[LV (Logical Volume)] | 由若干逻辑块组成（不要求逻辑或物理地址连续），每个卷可拥有各自的文件系统，每个卷的逻辑块对应的物理块可分布于不同的物理磁盘上，实现弹性存储。 |

逻辑卷和逻辑分区很相似，都可以对整个磁盘划分得到多个分区并分别运行不同的文件系统，但逻辑卷的强大之处还在于实现弹性存储。即既可以将多个物理磁盘构成一整个地址连续的逻辑卷，也可以将位于不同物理磁盘上空闲的物理块拼接成一个地址连续的逻辑卷。逻辑卷可以随便命名，而一般的分区都是/dev/sda1之类的。

# Virtual File System：虚拟文件系统层

# File System Layer：文件系统层

# General Block Device Layer：块层

# Driver Layer：驱动层