VFS机制

1. 《linux内核设计与实现》<第13章：虚拟文件系统>
   1. 特点
      1. VFS使得用户可以直接使用open（），read（），write（）等系统调用，无需考虑具体文件介质和实际物理介质。
      2. VFS在底层文件系统接口上建立了一个抽象层，提供了一个通用文件系统的模型（包括任何文件系统的常用功能集和行为）。
      3. 文件系统的控制信息存储在超级块中。
      4. 文件相关信息，即元数据，被存储在单独的数据结构中，称为索引节点（iNode）。
      5. 命名空间：文件系统被安装在一个特定的安装点上，该安装点即为命名空间，每个进程都有指定的一个唯一的命名空间（每个进程都会继承父进程的命名空间【除非特别声明】，所以所有进程往往只有一个全局命名空间）。
   2. VFS的四个主要对象类型
      1. 超级块对象
         1. 代表一个具体已安装文件系统。
         2. 该对象用于存储特定文件系统的信息，通常对应于存放在磁盘特定扇区中的文件系统超级块或文件系统控制块。
         3. S\_op 指向超级块的操作函数表；超级块对象由super\_block结构体表示；超级块的操作函数表由super\_operations结构体表示。
         4. 调用时候，把超级块作为参数传递，如 sb->s\_op->write\_super(sb);
      2. 索引节点对象
         1. 代表一个具体文件
         2. 包含了内核在操作文件或目录时需要的全部信息。
         3. 索引节点对象由inode结构体表示；索引节点对象由inode\_operations结构体表示。
      3. 目录项对象
         1. 代表一个目录项，是路径的一个组成部分。
         2. 目录项对象释放后也可保存到slab对象缓存中，此时，任何VFS或文件系统代码都没有指向该目录项对象的有效引用。
         3. 目录项缓存：

A．“被使用的”目录项链表。链表连接相关的索引节点，一个给定的索引节点可能有多个链接，所以就可能有多个目录项对象，因此用一个链表链接他们。

B．“最近被使用的”双向链表。 包含未被使用和负状态的目录项对象。总是在头部插入目录项（链头节点数据比链尾的数据更新）。

C．散列表和相应的散列函数。用来快速地将给定路径解析为相关目录项对象。

* + - 1. VFS把目录当文件对待，每个dentry代表路径中的一个特定部分；目录项也可包括安装点。
      2. 目录项对象并非真正保存在磁盘上，所以目录项结构体没有是否被修改的标志（是否为脏，是否要回写磁盘）。
      3. 目录项包括三种状态：被使用，未被使用，负状态。

A．被使用：它正在被VFS使用并且指向有效数据，不能丢弃。

B．未被使用： VFS当前没有使用，但该目录项对象仍然指向一个有效的对象，而且被保留在缓存中以便需要时再使用它。但若要回收内存，可以撤销未使用的目录项 。

C．负状态：没有对应有效索引节点，索引节点已被删除了，或路径不再正确了，但是目录项仍然保留，以便快速解析以后的路径查询

* + 1. 文件对象
       1. 代表由进程打开的文件。
       2. 该对象由相应的open()系统调用创建，由close()系统调用撤销。
  1. 其它
     1. 与文件系统相关的
        1. File\_system\_type: 描述各种特定文件系统类型
        2. Vfsmount： 描述一个安装文件系统的实例。
     2. 与进程相关的
        1. file\_struct
        2. fs\_struct
        3. namespace结构体